



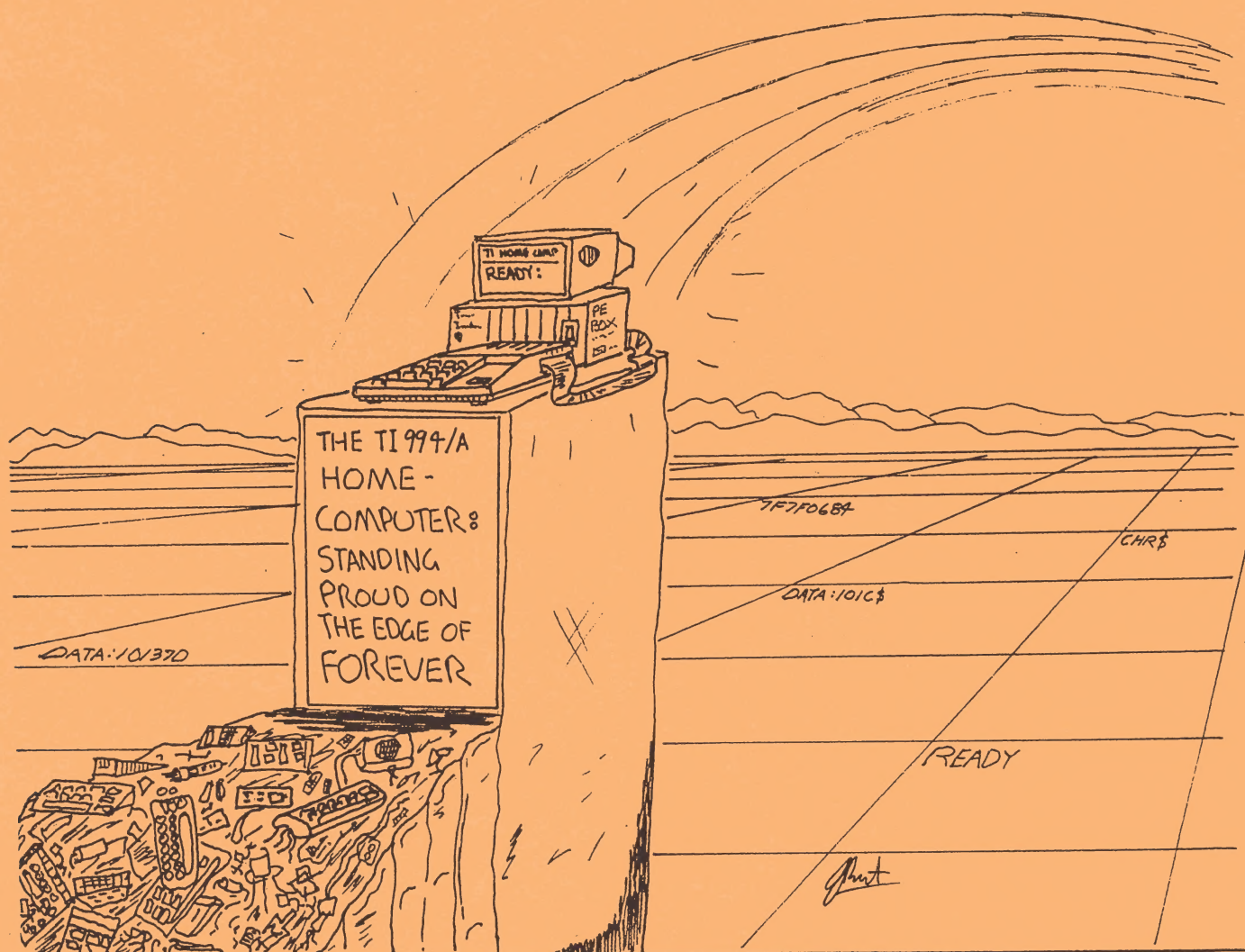
NEWS DIGEST

Focusing on the TI-99/4A Home Computer

Volume 7, Number 10

November 1988

Registered by Australia Post - Publication No. NBH5933



TIsHUG (Australia) Ltd.

TIsHUG News Digest

ISSN 0819-1984

TIsHUG News Digest

Index

November 1988

All correspondence to:

P.O. Box 214
Redfern, NSW 2016
Australia

The Board

Co-ordinator

Chris Buttner (02) 871 7753

Secretary

Terry Phillips (02) 797 6313

Treasurer

Percy Harrison (02) 808 3181

Directors

Cyril Bohlsen (02) 639 5847

Russell Welham (043) 92 4000

Sub-committees

News Digest Editor

Geoff Trott (042) 29 6629

BBS Sysop

Ross Mudie (02) 456 2122

Merchandising

Steven Carr (02) 608 3564

Publications Library

Warren Welham (043) 92 4000

Software library

Terry Phillips (02) 797 6313

Technical co-ordinator

John Paine (02) 625 6318

Regional Group Contacts

Carlingford

Chris Buttner (02) 871 7753

Central Coast

Russell Welham (043) 92 4000

Coffs Harbour

Kevin Cox (066) 53 2649

Glebe

Mike Slattery (02) 692 0559

Illawarra

Bob Montgomery (042) 28 6463

Liverpool

Larry Saunders (02) 644 7377

Northern Suburbs

Dennis Norman (02) 452 3920

Sutherland

Peter Young (02) 528 8775

Membership and Subscriptions

Joining fee \$5.00

Annual Family Dues \$25.00

Overseas Airmail Dues AUS\$50.00

or £22.00

or US\$30.00

Publications Library \$5.00

Texpac BBS \$5.00

Cover by Joshua Rust

TIsHUG Sydney Meeting

The next meeting will be at 10 am on 5th
of November at Woodstock Community
Centre, Church Street, Burwood.

Printed by

The University of Wollongong
Printery

Title	Description	Author	Page No.
Error codes	Software hints		28
Extended business graphs	Software review	Ryan, John	22
Extended display package	Software review	Sheehan, Craig	5
File types for TI99/4A	Software hints	Hott, Irwin	29
Forth column	Forth forum <7>	Smyth, George L	27
From the bulletin board	Mail to all		4
Games information	General interest	Brown, Robert	13
Games information	General interest	Judd, Stephen	13
Geneve 9640 an overview	General interest	Christensen, Garry	19
Joys of building your hardware	Cartoon	Amadio, Lou	8
Letter to editor	Comments on TND	Bowmer, Wade	3
Letter to editor	Congratulations	Andersen, Shane	3
Letter to editor	Databotics	Harris, DN	3
Listing BASIC programs	Software hints	Shaw, Stephen	20
Program to type in	Blackbox	Shaw, Stephen	15
Program to type in	Bowls	McMath, WA	14
Questprobe with spiderman	Adventure	Saunders, Larry	24
Rambles from Great Britain	General interest	Shaw, Stephen	23
Reading the disk bit-map	Software hints		21
Regional group reports	General interest		31
Secretary's notebok	Club news	Phillips, Terry	2
TI-Writer database	Word processing	Shaw, Stephen	6
TIsHUG shop	For sale	Carr, Stephen	3
TIsHUG software column	Club software	Phillips, Terry	2
Techo time	64K of 16 bit memory	Amadio, Lou	9
Techo time	64K of 16 bit memory	Trott, Geoff	9
They're off	General interest	Trott, Geoff	1
Tips from Great Britain	Software hints	Shaw, Stephen	11
Tips from the tigercub #51	Software hints	Peterson, Jim	25
Wire accessory I/O controller	Hardware project	Mudie, Ross	7
Younger set	Programs, adventures	Maker, Vincent	18

They're off

by Geoff Trott

Well we are just entering a busy time of the year for me and probably many of you also. As Terry points out in his article, the AGM is not too far away and everyone should start thinking of ways to keep the organisation running as well as and hopefully better than it has in the past. I have a worry that people who give their all in the running of TIsHUG do something like "burn out" when they relinquish their office. How many of the workers in past years are still active members and how many life members are still active? Even amongst our regional group we have had drop outs after intense effort on behalf of TIsHUG. Does the organisation create a too intense commitment for individuals who then cannot bear to go back to being just a member? It seems to me to be a pity to lose the people who have done their utmost for us all and who have experience which cannot be replaced.

continued on page 31

Secretary's Notebook

by Terry Phillips

Having just returned from Brisbane and Expo 88, I am attempting here to put together a bit of information that may be of interest to members from mail that accumulated in the post office box during my absence.

A response received from Dennis Faherty of Inscobot has put forward a proposition that has been accepted by the Directors, whereby our group will act as a dealer for the range of software marketed by Inscobot. Current titles and dealer/retail prices (\$US) are:

TI-Artist	\$11.95/\$19.95
Display Master	\$ 8.95/\$14.95
Artist Extras	\$ 3.85/\$ 6.45
TI-Base	\$14.95/\$24.95

Under the terms of the agreement, INSCEBOT will supply all documentation and a master disk of each program which will be duplicated locally. Final prices of this software range to members is not yet finalised but will be worked out when shipping costs, customs and sales taxes (if any) have been taken into account. An initial order has been placed and with a bit of luck it should be here in time for the November meeting. If anyone wants to reserve a copy of any of the above then they may give me a call.

Member Wade Bowmer has written in support of the decision to support fairware authors by collecting the requested donation on their behalf. Wade is the author of a program called Graphic Designer which will be released on an upcoming tape and disk.

Geoff Warner, Secretary of the TI Users of Perth has written thanking this group for its support in his endeavours to rekindle interest in the TI99/4A in the west.

New newsletters received and passed onto the publications librarian are:

TI-UP TITBITS (Perth)	- September
ROM Newsletter	- September
Ottawa Newsletter	- October
Sacramento News	- September
ATTIC (Adelaide)	- September
BUG BYTES (Brisbane)	- September
Hunter Valley	- September

Several others were also received prior to my holidays and passed onto Sysop, Ross Mudie, however I omitted to record their titles prior to handing them on.

Welcome is extended to the following new members who have recently joined the group:

Darren Watkins of Blacktown. Darren was previously a member and has now rejoined.
Ernest Lewin of Mooroolbark (Victoria).
Colin Christensen of Redcliffe (Queensland).
Colin is the Treasurer of the Brisbane Group, and also lives in the suburb where I have fond memories of the hotel and restaurant where our contingent to the Brisbane TI Faire had a great night out.

Michael Keane of Oyster Bay. Michael was also a former member who has now come back to the fold.

During my holidays I have had the opportunity (and the time) to have a play around with TI-BASE, and the entire membership records now reside on a file created from it. I am by no means an expert with TI-BASE, and with Shane Ferret's help, I hope to be in a position to use it as the definitive membership record keeping medium. Currently I store the membership records on a data-base created with SYMPHONY which runs through the Olivetti M24 which I use very regularly at my work. I have found Symphony's sorting capabilities the main reason for my use of it. It can actually sort my membership file in a matter of about 10 seconds with any field I care to sort it on.

Enough of these ramblings for this month. More news of interest (I hope) next month.

TISHUG Software

Column by Terry Phillips

From all the reports I have received, the October software copy day proved very popular. I suppose the good news is that only one disk from the software library was wiped out! Cyril was it really you who did this?

Not much to report on new software arrivals as it has been a rather quiet month, but those received are:

LIBRARY DISK A270 - A disk from Jim Peterson which contains D/V80 format files on the contents of his public domain library, a very useful PRINTALL utility as well as his Tigercub Tips, #46 through #54. A copy of this disk has been passed to the Editor, Geoff Trott, and no doubt the Tips will be featured in forthcoming issues of the TND. Jim has also requested that his library content files be passed onto members, so if you would like a copy please let me know. The disk is double sided and runs to 708 sectors in total. The library content files amount to 208 sectors.

LIBRARY DISK A271 - An exchange disk from the Brisbane User Group which contains a Bingo Numbers routine, an updated Disk Aid plus a Turbo Word Search program plus a couple of other small utility programs. Mostly written in assembly language. 310 sectors.

LIBRARY DISKS A272, A273, A274, A275, A276 and A277 - 6 floppy disks which contain Volume 1/1 to 1/6 of Barry Travers GENIAL TRAVELER, diskazine. There is some excellent general information on these disks which is sure to be of interest to the majority of members. These disks have been placed in the library, but will not be copied for general distribution. They will be available for demonstration purposes at the November meeting, then I propose they be passed around the Regional Groups for further demonstration purposes.

LIBRARY DISK A278 - ARCHIVER V3.02, an enhanced version which allows archiving/compressing in the one operation. 90 sectors and requires Extended BASIC and 32K and disk drive(s).

LIBRARY DISK A279 - FUNNELWEB V4.12, the latest version from the McGoverns.

Member, Bill Longmuir, has sent me a tape which he calls his Communications Package. It contains 3 programs; Morse Code, Signalling Flags and Signalling Pennants. Bill hopes to add further to this package as time permits. At the time of writing I have not had the opportunity to load and view Bill's work.

Software to be released at the November meeting will comprise 4 disks and a description of each follows.

LIBRARY DISK A209 - CHEQUE BOOK AND CREDIT CARD MANAGER, written by Richard Terry of the HV99er's, and as the title suggests this utility will keep you in touch with your expenses, and let you know where all your money has gone over the year. The program requires Extended BASIC and 32K expansion and a disk drive. This utility is released under the Fairware concept but the author has not placed any suggested donation in his documentation. If you use and enjoy this program please send an appropriate donation to the author.

LIBRARY DISK A260 - A disk from Germany which contains 3 utilities and a game all in assembly. Titles are Graftic-Master, 3D-World, Windos and Hopscotch. Load this disk through option 3 of Editor/Assembler, DSK1.LOAD. The game, Hopscotch, is similar to Q-Bert type games, but be warned the little fellow (kangaroo) does know only one word when dinged by one of the bouncing balls. Excellent graphics are a feature of this disk.

LIBRARY DISK A262 - This disk also contains a mixture. Rapid loaders for Infocom adventures, a great strategy game called Perfect Push, and a file to solve Hitchhikers Guide for all those who may be frustrated with that program. You will need Extended BASIC and 32K plus disk drive and for Perfect Push the Editor/Assembler module.

continued on page 19

Letters to the Editor

Once again, congratulations on a superb publication. Please note for your records, that I have moved and can now be contacted as follows...

Postal address: PO Box 640, Miranda, 2228
BLACKBOARD BBS (02)525 6970 (I am Sysop)

Keep up the fine work, it is appreciated.
Regards, Shane Anderson

Dear Editor and Readers,

I have some shopping information regarding Databiotics. I ordered a Desktop Publisher cartridge from them, and by rights if one orders in January and they acknowledge in February and say they are shipping in March, one expect action by June. Alas, it is now September! I was hoping by this time to flourish art done by this fabulous cartridge, which enables Cassette saving of artwork. It also accesses 32K memory expansion. I could save much longer bits of text and graphics with such a system, but for the fact that something appears to have happened. There was a bit of news about container vessel accidents, also there is reputedly a bit of pilfering on the wharves. Whatever the facts, perchance one could inquire what the trouble is. I have twice, since written to Databiotics and now as promised I am putting the problem into the TND.

If one pays \$100 to cover shipping, and handling plus cost of item, perchance it could be shipped airmail. There was no mention of any change, or of any extra money wanted. At this point it looks as if somebody grabbed it and ran!

It might be good for a few members to ask Databiotics what the trouble is! I cannot show you what a Desktop Publisher cartridge can do if the company does not get the thing to me!

Respectfully, D.N. Harris

Dear Geoff,

The work you have done for the Digest is excellent. From the headings to the editing, you are doing some fine work.

Now for the gripe: page numbers. Experience has shown me that the best place for page numbers is on the outside lower corner. To keep the same amount of space for the articles, I suggest a box for them in place of the border corner on the outside bottom.

Also I would like to see Tigercub Tips #40. Why? Because in Tips #47 (July 1988), Jim Peterson mentioned BXB (in Tips #40!) along with a modification which allows CALL CHAR(30,"") and CALL CHAR(31,"")!!

I would also be willing to do a short talk on my Tidbits at one of the Sydney Meetings in the new year (Chris Buttner please note).

Once again, you are doing a great job!

Yours, Wade Bowmer

The Editor replies

Thank you Wade for your letter. It is always good to get some response back from the readers. I think you are right about the page numbers, and if you look at the early issues you will find that we were doing just that. However, we are using a sheet with a border on it and when the printery reduce it they find it difficult to provide adequate borders unless we cut the bottom box off or they reduce the size more than we want. We decided to remove the bottom box from the sheet and so the page numbers had to go at the top. There is the TISHUG symbol on the top left of each page so that meant the page numbers ended on the top right of all pages. In a few months we will have used up all the paste up sheets and will need to design a new one. Then we will provide space for the page number at the bottom as well as for the month and year, which I like to see on each page as if someone copies a page the source is always there for all to see.

We would have printed Tigercub Tips #40 if we had them. I guess we will have to ask Terry to ask Jim Peterson for those that we have not printed (numbers 29 to 42 I think) and then we can do as you ask.

Meanwhile you should be able to find the one you want in the publications library in an overseas newsletter. As well as giving a talk, perhaps you would consider providing the occasional article as well. Finally, on behalf of Rolf and myself, thank you for your kind remarks. We only try to do our best.

TISHUG Shop with Stephen Carr

Welcome to my first report to you all. I have taken over the shop from Bob Bunbury who due to work commitments, has had to give up the position, so at least until the AGM in 1989 I shall try to serve you all.

The shop no longer has the following items:-
P-PUFF Printer Buffer Kits.

Clock Kit Parts.

But we will have at the next meeting a new item for sale:-

RAMdisk EPROM Operating System V1.0

This item contains an EPROM with the latest ROS, a disk with a menu program, BOOT, CHARAI, the latest DM1000 and finally installation instructions and how to use it. It will sell for \$20. We have only 11 to sell at first. Hopefully if more than 11 are sold and you would like to get one, you may place your name with me at the meeting or via TEXPAC BBS System, addressed to SHOP.

We have just received the latest Micropendium. Just to give you a taste of what is in this issue:-

Regena on BASIC

Demystifying Assembly

c99 Wipe-Out

Geneve 9640

TI Video Chips Reviews (2)

Newsbytes

User Notes

Classified

and that is still only \$3 at the shop or can be posted to you for an extra \$0.50. We still have, while stocks last, the shop's special on back issues of Micropendium Magazines, prior to January 1988 for \$0.50 each. There are also 3 Clock PCBs at \$2.

All the regular items will be there but if there is something that you want and we do not have it in stock, please be patient and we will try to get it for you, either via the mail, hand delivered, or at the next meeting.

That is all for now and see you at the next meeting.
Steven Carr

For Sale

PE box, complete with 32K memory expansion, AT multifunction card, twin disk drives (one requires 110 volt ac supply)

Speech synthesizer

TI-Writer complete with manuals

TI Multiplan complete with manuals

TI Logo and curriculum guide with manuals

Extended BASIC module and manual

The following modules

Personal Record Keeping

Personal Report Generator

Disk Manager 2

Addition and subtraction 1

Addition and subtraction 2

Division 1

Division (Milliken)

Music Maker

Assorted club disks

Video Chess

Parsec

Terminal Emulator

Original black and silver console, power supply, UHF modulator etc.

Total price, \$900 or nearest offer. I am prepared to split the system if there are no takers for the complete system.

Contact David Smith on (069)26 1342

From the Bulletin Board

MAIL TO : ALL
MAIL FROM : SYSOP

Attention members needing TMS9900 or TMS9901 chips.

TIshUG is purchasing a quantity of each of these chips contact Cyril or SHOP for one off prices.

For people wanting quantities of these contact Peter Gleed in Melbourne on (03)877 4790. Prices are approximately \$10 each for TMS9900 + packaging and postage.

Colour Monitors.

Peter Pedersen is interested in organising a bulk purchase of colour monitors. He needs at least 4 other people to get a bulk discount price of \$180 each.

These are reconditioned units and will require a special interface between the TI99/4A modulator socket on the console and the monitor.

Interested? Contact Peter Pedersen on (02)772 2396

MAIL TO : ALL
MAIL FROM : GOWFAR

Hi to all! Yet another correction coming to you from "THE CORRECTOR!" Seen in the October 1988 issue of the TND, a review of 4A/Talk, by a yank, Scott Darling "SYSOP of GENIE, USA". You would think he would know better, eh? At the end of his article, he said that he slightly downrated 4A/Talk as, amongst other things, "there is no way to dump a buffer captured file when it is full" and "there is no provision for sending all of an ASCII file, when logged on" (meaning as opposed to sending one line at a time with function D). As to the first bit, hitting CTRL[5] will bring up the DUMP menu, for you to take your choice of 1) Dump; 2) Erase; 3) Exit. Option 1 allows a dump to disk or PIO or RS232. Why did not that "SYSOP" know that? As to the second part, to send all of an ASCII file in one hit, you have to, first, hit FCTN[6] and choose option 3 to set up the XON and XOFF. Once this is brought up, set the XON to a different number to XOFF (preferred use is XOFF 17 and XON 19) and then hit enter. Then, open the keyboard file, hit FCTN[D] to load the first buffer full of data and hit it again to send the rest of the file, no matter what length it is. This "SYSOP" states that the 4A/Talk manual is well written and leaves nothing to the imagination. My question is: Did someone tell him that? He could not have actually read it! I hope this letter gets printed in the next TND as a correction to the mistake. 4A/Talk is, without doubt, the best TI99/4A terminal emulator program available.

MAIL TO : ALL
MAIL FROM : PETESAKE

The A.T. Music Card is nearing completion and kits of parts and PCB supplies will soon be announced, so I would like to hear from anyone interested in building this system. It will provide 16 channels of programmable sound into stereo outputs, and can be used on any system. Due to the limited number of sound chips only 18 kits will be offered, until further stocks are located. If you would like to reserve a kit get in touch with me on (02)358 5602, or place an order with Steven Carr at the club SHOP.

MAIL TO : ALL
MAIL FROM : SCIFI-BBS

Hi to all yet again. OK, it is official now. The Geneve will run my BBS program, therefore I am buying one. Hurray for me! However, news for you is that Gary Christensen, in Qld., will be placing the new order in about 2 weeks. If any of you are interested, Gary quotes the price of the machine at \$750, at your doorstep. I would like to hear from anyone who wishes to get one. We may be able to get together and order, with Gary, at once. Remember that the deposit, at least, will be called in about 2 weeks. I have no idea, yet, how much that deposit will be but be sure you have enough cash in the event that he asks for the whole lot in advance. SCI/FI BBS on a Geneve before the year is out (hopefully)!!

Regards, Greg.

MAIL TO : ALL
MAIL FROM : DUTCHY
To all

Supply of 62256LP12 Memory chips. NJS components have informed me that they hope to have 62256LP12 chips available late this week or early next week. Price is \$16.42 + 20% tax each. Phone number is (03)887 0577, contact name Steven Stantion.

Any questions please phone me W/H (02)359 2161. Keith de Graauw.

MAIL TO : ALL
MAIL FROM : LARRY

Picasso

The complete story by Asgard Software. A controversy has been raging throughout user groups over the status of the Picasso drawing program, by Arto Heino. Asgard hopes this clears up some of it.

Picasso 1.0 appeared on the scene almost a year ago in the US. It immediately become a very popular item on the BBS networks, where it was first distributed. User groups such as the LA99ers began distributing it immediately. Against this backdrop of the immediate success of a truly remarkable program, Chris Bobbitt of Asgard Software contacted the author in November to discuss the possibility of developing other products for the TI99/4A. The author described to Chris a new version (2.0), which was a vast improvement over the previous versions and offered Asgard rights to manufacture the program in North America.

In February, after contracts had been signed and promises exchanged, Asgard belatedly learned that significant portions of Picasso were taken from a program call "Paint 'n Print" by Navarone. The author then refused to answer letters. Asgard was in a quandary; they had paid a considerable advance to Arto Heino and had spent money on preparing the program for release, but seemingly the program not only could not be sold, but violated copyright laws, and the author has disappeared!

To make a long story short, Asgard contacted the current owners of the rights to "Paint n' Print" and worked out an agreement with them. Asgard manufactures the item and Tenex has the exclusive rights to sell it. Previous versions of Picasso may be distributed under the freeware method, however they are inferior to the commercially available version. Everyone benefits from this arrangement, the author gets a royalty from sales of his work, Tenex capitalizes on its rights to the original, user groups can still distribute version 1.1 as a trialware program, TI99/4A users can purchase a supported version of this very capable and useful program, and Asgard recoups its initial investment.

***** New Products

QUICK-RUN is an Extended BASIC utility which makes other Extended BASIC programs run instantly. PREDITOR is complete replacement for the Editor/Assembler or TI-Writer editor.

Both will be demonstrated at the October Liverpool meeting if they arrive in time.

Bye for now, Larry.

MAIL TO : ALL
MAIL FROM : LARRY

LEGENDS 1.1 has a bug in it. The error is in line 1200 of the file LGDN/MON. The line reads:
1200 OK=K :: DISPLAY AT(22,1):USING 9: "WESTERN PORTAL"
:: CALL W....

The bug is the first part of the line, OK=K. There is no variable K. Change it to read:
1200 OK=F :: DISPLAY AT(22,1):USING 9:
"WESTERN PORTAL" :: CALL W....

Note that Pre_Scan has a limit of 100 lines. If the program has more than 100 lines, change the DIM in the program or it will crash out when it hits 101 lines. Look for variables and if program has some do not change, if no variables in program then say yes to change variables when running Pre_Scan.

Bye for now LARRY.

Extended Display Package

by its author, Craig Sheehan

In the second part of this series we will examine the ACCEPT command in depth, as well as forty column mode and scrolling briefly. As always, before using XDP, the utility must be loaded. To do this:

1. From Extended BASIC, insert the XDP disk in drive one.
2. Type in: RUN "DSK1.XDP"
3. Type in: CALL LINK("NEW")

XDP's accept command is, like DISPLY, a more advanced version than Extended BASIC's equivalent. It allows multi-line inputs, validation of input and the setting of enter keys and the cursor's starting position. Further, it is possible to have the enter key pressed and the position of the cursor returned to the program. The command has the format:

```
CALL LINK("ACCEPT",row,column,specification string,
return variable[,variable list])
```

Basically, 'row' and 'column' refer to the screen position for the start of the input field. The 'specification string' contains specification commands that are similar to those we examined last month in the DISPLY command. The optional variable list is utilised by the specification commands, and the return variable is where the data entered is returned when an enter key is pressed. The specification string may be up to 150 characters long and contain specification commands which, as we learnt last month, consist of two characters preceded by a period with arguments sometimes following. Before examining the example program, we will build up another example to demonstrate the specification commands used. We will consider a program segment to enter a file name. A no frills version is shown below:

```
100 CALL LINK("XDP")
:
500 CALL LINK("DISPLY",12,1,"FILE NAME?.CRDSK1.")
510 CALL LINK("ACCEPT",13,1,"",FILENAME$)
520 ! REST OF PROGRAM
```

With the specification string left blank as above, all of the routine's defaults take effect. The default conditions are an input field to the bottom of the screen or 255 characters, which ever is the least, a key unit of 5 (i.e. normal keyboard) and all characters valid with the cursor beginning at character 1. 'ENTER' is the only enter key defined under default conditions, so the up and down arrow keys can not be used. The first variation to these default conditions that you may want to make is to limit the size of the input field. This is changed using the '.SI' command. The argument following it determines the size of the field. Like Extended BASIC's ACCEPT command, a positive value causes the input field to be cleared with blanks before data is entered, and a negative value uses the characters on the screen as a default input.

For example,

```
510 CALL LINK("ACCEPT",13,1,".SI(-25)",FILENAME$)
```

would allow a maximum field length of twenty five characters, with the characters on the screen being used as the default input.

In a similar way to the DISPLY command, '.VA' can be used to obtain a value from the variable list:

```
510 FLDLEN=-25 :: CALL LINK("ACCEPT",13,1,".SI(.VA)",
FILENAME$,FLDLEN)
```

would have the same result as the previous example.

Several specification commands are used to emulate the VALIDATE feature of Extended BASIC. '.DI', '.NU' and '.UA' replace digit, numeric and alpha respectively. Another two options have been added: '.LA' to validate lowercase alpha keys and '.VS' to validate the space bar. To make keys valid other than

those covered by the above commands, the '.VL' command is used. After adding the validate commands to our example, it becomes:

```
510 CALL LINK("ACCEPT",13,1,".SI(-25).UA.DI.VL(.)",
FILENAME$)
```

This would allow all upper alpha keys, the digits 0 to 9 as well as the period to be accepted as valid input. If you wish to validate the quote or a closing parenthesis with the '.VL' command, then it must appear as a double character (e.g. '.VL(")"))' would validate the quote and closing parenthesis character). Also note that the '.VA' option can not be used with '.VL(...)'

One trick often used in Extended BASIC is to execute a CALL KEY(3,K,S) prior to an accept so as to convert any lower case keys pressed to upper case. Since XDP's accept command resets the key unit to 5 each time it is called, this trick will not work. The way to overcome this is to use the specification command '.K3' which sets the key unit to 3. Note that it is not changed back to 5 upon return to Extended BASIC. Adding this command changes our example to:

```
510 CALL LINK("ACCEPT",13,1,".SI(-25).UA.DI.VL(.).K3",
FILENAME$)
```

The starting column for the cursor may also be specified. In the case where the prompt is "DSK1.", it is often more convenient for the user if the cursor begins after the period. By adding the specification command '.CP(6)' this is achieved. The '.CP' command defines which column of the input field the cursor will begin on, which in this example is the 6th character. The use of '.VA' in '.CP' is supported and if the corresponding value is a numeric variable, the column the cursor was on when an enter key is pressed will be returned in this variable.

So, after all of the above modifications are made, our glorified filename input routine becomes:

```
100 CALL LINK("XDP")
:
500 CALL LINK("DISPLY",12,1,"FILE NAME?.CRDSK1.")
510 CALL LINK("ACCEPT",13,1,
".SI(-25).UA.DI.VL(.).K3.CP(6)",FILENAME$)
520 ! REST OF PROGRAM
```

This will allow you to enter a file name, with the cursor beginning at column 6. Lower case keys will be converted to upper case and only the characters required to form a filename are valid.

A practical application of the accept command is as a line editor. The program below shows how to implement this, with the editor subprogram being on lines 10000 to 10100. Most of the action is centered around the accept command on line 10030. For correct operation, an array must be created in the main program and the number of elements placed in a variable that is passed to the editor. Forty column mode must also be invoked. One way to doing this is to place the numeric value '40' as the first argument of the XDP command, as shown on line 110. Line 120 simply calls the editor and 130 returns screen handling to normal Extended BASIC commands. Before typing in the program, ensure that XDP has been loaded and the memory cleared with CALL LINK("NEW")

```
100 DIM TEXT$(50):: MAXLINES=50 ! MAXLINES >=24 and
must equal the subscript value of TEXT$.
110 CALL LINK("XDP",40)
120 CALL EDIT(TEXT$( ),MAXLINES)
130 CALL LINK("NORMAL")
10000 SUB EDIT(TEXT$( ),MAXL):: CALL LINK("CLRS") :: FOR
ROW=1 TO 24 :: CALL LINK("DISPLY",ROW,1,TEXT$(ROW))
:: NEXT ROW
```

```

10010 ROW,COLUMN,FIRSTL,EKP=1
10020 AID=1 :: DOWNARROW=10 :: UPARROW=11 :: ENTER=13
      :: BACK=15
10030 CALL LINK("ACCEPT",ROW,1,
      ".SI(-40).CP(.VA).EK(1,10,11,13,15,.VA)",
      TEXT$(ROW+FIRSTL-1),COLUMN,EKP)
10040 IF EKP=BACK THEN SUBEXIT
10050 IF EKP=ENTER THEN COLUMN=1 :: EKP=DOWNARROW
10060 IF EKP=UPARROW THEN ROW=ROW-1 :: IF ROW=0 THEN
      ROW=1 :: IF FIRSTL>1 THEN
      CALL SHIFTEXT(TEXT$( ),FIRSTL,1,-1)
10070 IF EKP=DOWNARROW THEN ROW=ROW+1 :: IF ROW=25 THEN
      ROW=24 :: IF FIRSTL<MAXL-23 THEN
      CALL SHIFTEXT(TEXT$( ),FIRSTL,24,1)
10080 IF EKP=AID THEN COLUMN=COLUMN+5 :: IF COLUMN>40
      THEN COLUMN=40
10090 GOTO 10030 :: SUBEND
10100 SUB SHIFTEXT(TEXT$( ),FL,ROW,DIR):: FL=FL+DIR ::
      CALL LINK("SCROLL",-DIR,0)::
      CALL LINK("DISPLY",ROW,1,TEXT$(FL+ROW-1)):: SUBEND

```

The editor itself is quite easy to understand (do not let the accept fool you!). Lines 10000 to 10020 are concerned with putting the text already entered on the screen and initialising the variables and some constants. Line 10030 forms the backbone of the program. It sets an input field of forty characters, using the characters on the screen as a default. 'CP' is used to set the starting position of the cursor, and since 'VA' is used, the value is obtained from the variable list. It is also important to note the value in the variable list for 'CP' is also a numeric variable. This will cause the position of the cursor when an enter key is pressed to be returned to the corresponding variable.

Finally the enter keys are defined using the 'EK' command. The numerals 1, 10, 11, etc. correspond to the code returned by CALL KEY of the desired enter key. Again the 'VA' option is used to return the code of the enter key pressed to the program.

The rest of the program is concerned with taking the appropriate action for particular enter keys. Consider as an example the <enter> key. If this key is pressed, the variable 'EKP' returns a value of 13. What must be done is obvious: the cursor must be moved down a line, scrolling the screen up one line if the cursor was on the bottom row, and putting the cursor in column 1. This has been achieved by setting COLUMN=1 and then changing the value of EKP to the code of the down arrow key. The down arrow routine works by adding one to the row and then determining if text needs to be scrolled. If it does, the text is scrolled up one line and new line placed at the bottom of the page. As an exercise you may work out how all of the other key functions are implemented, and maybe add more functions.

Line editor keys:

Function:	Key:	EKP
Tab	FCTN[7]	1
Down Arrow	FCTN[X]	10
Up Arrow	FCTN[E]	11
Enter	<ENTER>	13
Back	FCTN[9]	15

References from the XDP Utility Guide for the new commands mentioned in this months article are ACCEPT (pages 12-14), CLRS (page 17), NORMAL (page 26) and SCROLL (page 32).

XDP is available from the club shop in either double sided or floppy disk formats. Questions related to the package may be directed to me via the address given in last month's article or under the table of contents in the XDP Utility Guide.

Next Month: Windows and how to use them.

A tutorial on using XDP from the beginning will be presented at the November full day workshop. Participants need only a minor knowledge of Extended BASIC programming as well as a note pad and pen. A disk containing example programs will be available from the club shop on the day.

TI-Writer Database

by Stephen Shaw, UK

TI-Writer can function as quite an effective database, if you use FS to locate a specific item. You can even use M to put your items into order.

The program which follows will enable you to use each line of TI-Writer as a separate record, divided into up to 9 fields, and with up to 400 records. Not bad eh! Quite sufficient for simple database purposes.

This program is an Extended BASIC sorting routine, which will sort the records by any one field.

The fields are identified in line one of the TI-Writer file, by numbers located in the left-most column of each field. Numbers may be zero filled, or RIGHT justified, as follows:

```

1      2      3      4      5
TI*MES JAN-MAR 88 43 PILOT 99 ARTO HEINO
TI*MES JAN-MAR 88 8  DISK DRIVES STEPHEN SHAW

```

Save the DV80 file using SF as usual, this program will strip the tab records out for you!

Use of TI-Writer as a database is of value as it is familiar, easy to use, and easily edited!

This program comes from COMPUTER BRIDGE, Vol 5 No 11, Nov 86, and was written by George Paschetto.

```

100 REM PRESCAN
110 GOTO 120 :: D$,X,Y,K,S,Z,FN,FS,FL,L,S$,H,J,B$ ::
      CALL KEY :: CALL SOUND :: CALL HCHAR
120 DISPLAY AT(4,2)ERASE ALL:" See Instructions? Y/N"
      :: DIM A$(400),F(10)
130 !@P-
140 ACCEPT AT(5,12)VALIDATE("YN"):D$ :: CALL CLEAR ::
      IF D$="N" THEN 230
150 PRINT " This program was written to sort files
      created with the TI-Writer. The file can be as
      long as 400 lines."
160 PRINT " The first line of the file must contain the
      field numbers, like this:"
170 PRINT "1 2 3 <-first line:"102 NJ Paul
      Peters:"314 MO Carol Corrina:"622 AL Hu Noes":
180 PRINT " You may have up to 9 fields but they must
      all fit on one line.": :: INPUT "<press enter>":D$
      :: CALL CLEAR
190 PRINT " The lines can be up to the full 80
      character s long that TI-Writer supports. This
      program will only sort one field at a time."
200 PRINT : " The sort is always":"ascending order,
      with the lowest value first."
210 PRINT " The file can be used by TI-Writer after
      this program is through with it.":
220 INPUT "<press enter>":D$ :: CALL CLEAR
230 DISPLAY AT(12,1):"Input device and file name:":
      DSK "
240 ACCEPT AT(13,5):D$ :: IF D$="" THEN 240 ELSE OPEN
      #1:"DSK"&D$,DISPLAY,VARIABLE 80,INPUT
250 CALL CLEAR
260 REM LOOK FOR 1ST LINE
270 LINPUT #1:A$(0) :: IF EOF(1) THEN 610 ELSE IF
      A$(0)=" " THEN 270
280 FOR X=1 TO 400 :: LINPUT #1:A$(X) :: IF EOF(1) THEN
      310
290 NEXT X
300 REM F( )=field's positions (count down 2 from
      eof():last 2 lines are tabs)
310 X=X-2 :: CLOSE #1 :: FOR Y=1 TO 9 ::
      F(Y)=POS(A$(0),STR$(Y),1) :: IF F(Y)=0 THEN 330
320 NEXT Y
330 F(Y)=80 :: Y=Y-1 :: IF Y THEN 350
340 PRINT "Can't find field marker":"This was found
      instead.":A$(0) :: STOP
350 DISPLAY AT(10,5)ERASE ALL:"Press.":"1. To
      Sort":"2. To Save to disk":"3. To QUIT "
360 CALL KEY(3,K,S) :: IF (K<49)+(K>51) THEN 360 ELSE
      CALL SOUND(-20,880,0) :: ON K-48 GOTO 380,540,580
370 REM GET FIELD TO SORT
380 DISPLAY AT(1,3)ERASE ALL:"<choose field to sort>"
390 FOR Z=1 TO Y :: DISPLAY AT(Z*2,1):"Field No.":Z;
      " "; SEG$(A$(1),F(Z),F(Z+1)-F(Z)) :: NEXT Z
400 CALL HCHAR(20,1,30) :: CALL KEY(3,K,S) :: FN=0 ::
      CALL HCHAR(20,1,32)

```

continued on page 22

Wire Accessory I/O Controller

Demonstration for TISHUG Tutorial Day, 5/11/88
by Ross Mudie

1. The Task.

The directors of TISHUG asked me to put together a small train set being controlled by a TI99/4A computer, using the Wire Accessory I/O Controller which I developed and showed at the June 1988 tutorial day. The project which is now operational has taken 18 days, so far, of intense spare time effort with the assistance of younger son, Peter. The task has taken the following stages:

a) Expansion of the prototype Wire Accessory I/O from 8 inputs and 8 outputs to 24 inputs and 32 outputs.

b) Development and construction of a small HO gauge train layout comprising an oval shaped track, passing loop and terminating siding. The layout contains 3 points and 6 signal stands.

c) Design, development and construction of track detectors which allow the position of a train to be monitored in addition to control of track power, signals and points.

d) A watchdog circuit which stops the train set if the computer fails to operate.

e) The design and development of a train motor controller which can be operated from 4 outputs of the Wire I/O using binary code to control the pulse width of the pulsed 12 volt output to the train motor.

f) The development of an assembly program to respond rapidly to the needs of the train set in real time.

g) Development of a simple variation of the implanting techniques to allow the Extended BASIC linked assembly to be saved in an Extended BASIC program on cassette tape and then loaded into a 32K console.

This train set uses smart tracks and dumb engines. The points are operated via the computer keyboard, train speed is controlled from the key board or Joy Stick, track sections can be turned off from the keyboard and signalling is automatic.

2. Program function blocks.

The program is written in modular form with separate modules performing functions in a continuously scanning and operating loop. The inputs from the track detectors are placed in a block of memory called INBUF. Outputs from the program are placed in a block of memory called OUTBUF. Once in each program loop, the program scans the inputs from the I/O peripheral, then performs the other functions of the program and then transfers the outputs from OUTBUF to the I/O peripheral. The train layout is shown in a mimic display on the screen which shows train positions, point status, signals, train speed, stopped sections and Forward Hold status.

a) Input Scan.

Scans the I/O peripheral and places the results in INBUF. Using information provided by the voltage detectors, decisions can be made of train direction and a technique called Forward Hold can be used to prevent signal flickering when the train wheels momentarily leave the tracks by storing a train's position until it reaches the next track section.

b) Keyboard Scan.

Scans the keyboard and Joy Stick for valid inputs as follows:

D - Diagnostic mode.

E - Exit for D, H, R, S and V.

F - Forward Hold.

G - Exit from Forward Hold.

H - Help, shows key functions on screen

I - also Joystick Up, Increase train speed and go into forward after reverse.

M - also Joystick Down, decrease train speed and go into reverse after forward.

P - Pause.

R - Restore track voltage after S.

S - Stop train by switching track voltage in individual track sections.

V - Version of software.

1-6 Operate points.

FCTN[4] - Return to Extended BASIC.

All other keys are ignored, alpha lock can be in either position and a key held down is only actioned on the first scan in which it was detected with the exception of I, M and the joy stick which auto repeat at half the watchdog output rate.

c) Tests and Logical decisions.

This program module tests the inputs stored in INBUF and the point position flags to control track power, signals on both track and screen in addition to points which are about to change. Signals at a point go red prior to a point changing and points are prevented from changing whilst a train is in the point track section.

d) Points.

Checks the point flags and if permitted provides a timed pulse to the appropriate point by placing the required state in OUTBUF. The point pulse is approximately 200 msec so the output is simply set and then the program goes on around the loop. After the timing period, a number of loops later, the instruction to turn off the point is issued. To allow the capacitor in the point power supply to recharge a delay is implemented before another point can be changed. Point change requests are stored and only executed when permissible, one at a time.

e) Track Control.

This allows any number of sections of the track to be turned off or on again by keyboard command. When S is pressed all the track section numbers flash in their track sections. When a valid key is pressed the appropriate track number flashes alternately red/number to show the stopped sections. R followed by a stopped number Restarts a Stopped Section. E may be used to Exit the routine.

f) Diagnostic Screen Display.

Diagnostic shows the state of INBUF, OUTBUF and the control flags whilst the train set is actually operating. The track mimic display is on the left of the screen whilst the diagnostic is on the right of the screen. E will Exit the routine.

g) Signals on Screen.

Displays the condition of signals on the screen mimic display adjacent to the track.

h) Watchdog.

Provides pulses at just under one second period to hold up the hardware watchdog in the train set interface to indicate continued program operation and flashes a green watchdog lamp on the screen. If the hardware watchdog timer in the train set loses the regular pulses it turns all signals red and prevents operation of any points.

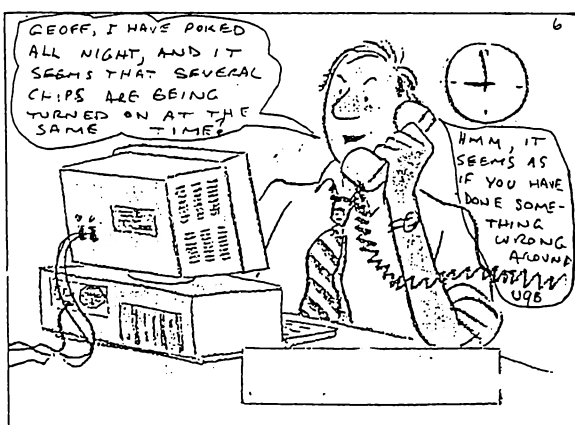
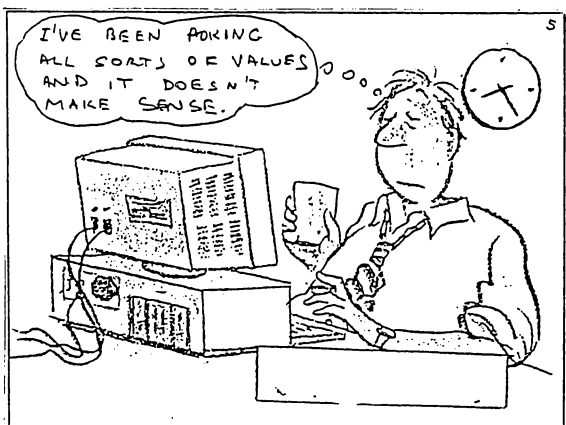
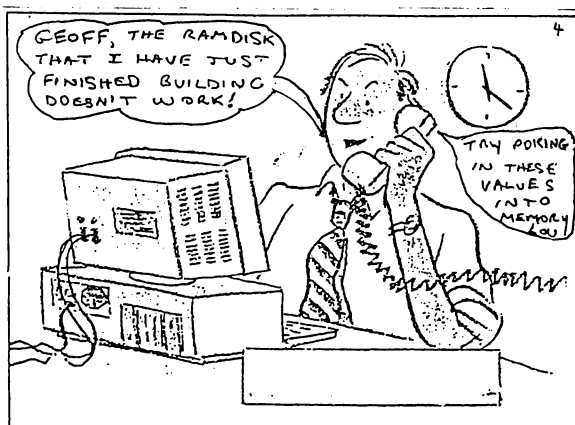
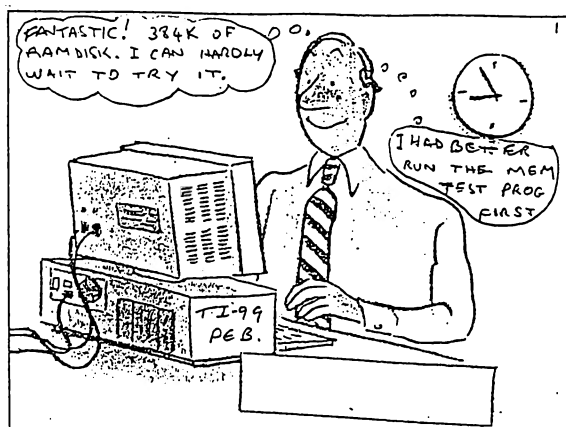
i) Output.

Transfers the contents of OUTBUF to the Wire Accessory I/O Controller.

j) Sound Off.

continued on page 20

The Joys of Building your own Hardware



Techo Time 64K bytes of Memory on the 16 bit bus

by Lou Amadio and Geoff Trott

Ever since I first read about the "16 Bit 32K Memory In The Console" by Ron Marissen about 6 months ago I have been curious to find out just how much difference it would make compared to the standard 32K as provided by TI (on the 8 bit data bus). The project, however, was put off for a long time for one reason or another.

Recently, an opportunity arose to do some hardware hacking, so I approached Geoff Trott. Geoff did not need much persuading as he enjoys the odd hardware project, especially after many long hours editing the TISHUG News Digest. With the availability of 32K byte static RAM chips, we naturally wanted to use these as they are a good deal cheaper (per byte) than the 8K chips which Ron Marissen used in his original version. Ron described how to install 4, 8K RAM chips over the 2 system ROM chips. Although we now had 32K in one chip we still had to use 2 chips as each host chip in the console (the system ROMs) were only connected to half of the data bus of the TMS9900 CPU.

Apart from the 32K chips (2 of), we also needed another control chip. Geoff settled for a 74LS21 (dual quad input AND gate). Where the original article called for a total of 7 chips, Geoff managed to do it with only 3 - this simplified the construction considerably. So in fact, we ended up installing 64K byte of static RAM of which only 32K byte were required by the CPU. We were wondering whether we could bank switch the remaining 32K bytes, but this would require special software to be written. Geoff then struck upon the idea of manually switching different banks of this "free" 32K memory within the normal address range of the CPU (see notes on enhancements).

In the process of installing the 32K chips, Geoff decided to remove the 6810 Console (scratch pad) RAM (256 bytes) in order to give better access to some PCB tracks and simplify the circuit. The scratch pad RAM would now be part of the "free" 32K and thus was quadrupled in size to a full 1024 bytes. The normally unavailable 768 bytes could now be used for some unique utility software requiring high speed RAM, such as interrupt routines or load interrupt routines. Routines using this area would not be corrupted by running any current program.

This is how we initially used the new static memory:

32K CPU RAM on the 16 bit data bus

Low memory >2000 to >3FFF

High memory >A000 to >FFFF

1K Console RAM on 16 bit data bus

1K bytes available at >8000 to >83FF

The above memory could have battery back-up if required.

How did it perform?

Improvements in speed were expected since access to the 32K was now done as a "WORD" (16 bits) rather than than as a "BYTE" (8 bits), and also due to the elimination of memory "wait states" which were incorporated by TI into the design to allow for slow access memory chips. With all timings in microseconds, reading a word or byte takes .667 instead of 2, while writing a word or byte takes 1.5 instead of 4.333.

It was found that power requirements of the console were marginally reduced with this modification (including 2 6810 RAM chips removed).

The following programs were superficially tested in order to quickly gauge the performance of the new memory expansion.

TI-Writer - approximately 30% faster for Replace String; approximately 50% faster for enter/delete functions.

Multiplan - approximately 30 % faster.

Parsec - no difference?

TI Runner - noticeably faster

Buck Rogers - noticeably faster

TI Artist - satisfactory

Munchman - noticeably faster

Computer War - too fast?

The following programs did not work with the 16 bit 32K, possibly due to the way that the programs access VDP RAM:

Tennis, Ant Eater, River Boat Rescue, Submarine Commander.

How to do it

The instructions and diagrams below describe how to install the new memory chips. This article is not intended to be a step by step guide and anyone not thoroughly familiar with hardware hacking should consult the advice of their local "techo".

Minimum parts required are: 2 of 62256, 32K x 8 bit static RAMs

2 of 24 pin IC sockets

1 of 74LS21, dual quad input AND gates

- 1) Locate and remove the system ROM chips (U610 and U611) on the TI99/4A mother board and solder IC sockets in their place. These chips are in sockets in order to facilitate removal of the 32K memory expansion at a later date, if required. A good solder sucker is recommended for this step to prevent possible damage to the PCB or the chips. (Note that it is possible to add the 32K without removing any chips, but extra chips will have to be installed and other tracks/pins cut.)
- 2) Remove and discard the two Console (scratch pad) RAM chips (U608, U609 - part number 6810). This step frees up an input to stop the wait state generator as well as allowing the expansion of the scratch pad RAM to a full 1K byte.
- 3) Desolder pin 8 of U507 from the motherboard, cut the base of the pin and bend it out for further connection. You may find it easier to completely remove this chip, bend out pin 8, then replace it.
- 4) Carefully bend the legs of the 62256 chips in a little (mind static electricity) so that they will sit firmly on top of the host chips. Bend out pins 1, 2, 20, 22, 23, 26, 27 and 28 of each 62256 prior to installation to facilitate wiring later.
- 5) Place one 62256 chip over U610 and the other over U611, facing them the same way as the host chip. Pins 1, 2, 27 and 28 of the 62256s will hang over the end of the ROMs. Solder as per instructions below. Do not forget to mark each set in some way so that the bottom chips (U610, U611) are inserted into their correct sockets on the mother board.
- 6) Bend legs 7 and 14 of the 74LS21 in a little and bend all other pins out for further connections. Locate this chip over U507 (facing the same way) and solder as per instructions below.

Make the following connections using thin gauge insulated single core wire where necessary:

Pin 1 of both 62256s to pin 3 of U504 (A0(H)).

Pin 2 of both 62256s to pin 1 of U504 (A2(H)).

Pins 3 to 14 of 62256s to pins 1 to 12 of host ROM (U610 or U611).

Pins 15 to 19 of 62256s to pins 13 to 17 of host ROM (U610 or U611).

Pin 20 of both 62256s to pin 8 of 74LS21 (CS(L)).

Pin 21 of both 62256s to pin 19 of host ROM.

Pin 22 of both 62256s to pin 11 of U602 or pin 9/10 of U508 (DBIN(L)).

Pin 23 of both 62256s to pin 6 of U503 (A3(H)).
 Pins 24 to 25 of 62256s to pins 22 to 23 of host ROM (U610 or U611).
 Pin 26 of both 62256s to pin 2 of U504 (A1(H)).
 Pin 27 of both 62256s to pin 16 of U608 (WE(L)).
 Pin 28 of both 62256s to +5 volts.
 Pin 1 of 74LS21 to pin 14 of U504 (>2000 page select).
 Pin 2 of 74LS21 to pin 10 of U504 (>A000 page select).
 Pin 3 of 74LS21 not connected.
 Pin 4 of 74LS21 to pin 9 of U504 (>C000 page select).
 Pin 5 of 74LS21 to pin 7 of U504 (>E000 page select).
 Pin 6 of 74LS21 to pin 9 same chip (expand the selection gate).
 Pin 7 of 74LS21 to pin 7 of U507 (host chip).
 Pin 8 of 74LS21 to pin 20 of both 62256s (for chip select) and pin 11 of U608 (to disable the wait states and 16 to 8 bit converter).
 Pin 10 of 74LS21 to pin 8 of U507 (scratch pad RAM select). Note: pin 8 of U507 should be disconnected from the mother board.
 Pin 11 of 74LS21 not connected.
 Pin 12 and 13 of 74LS21 to pin 14 same chip.
 Pin 14 of 74LS21 to pin 14 of U507 (host chip).

Enhancements

"The following enhancements were intended to make use of the available "free" 32K arising from installing 2 of 62256 chips inside the console. These modifications should only be carried out with caution. Depending on how the switching is arranged, the possibility exists for two lots of memory chips to be accessed simultaneously within the same address space. The results could be fatal! (to the chips that is!)"

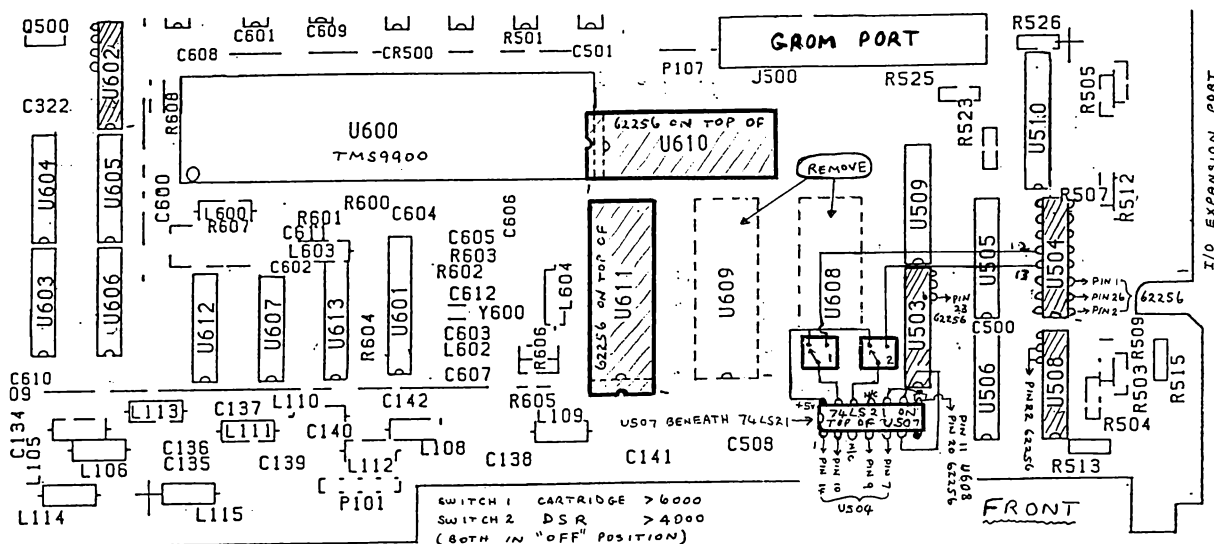
1) It is possible to invoke memory in the 8K cartridge space (>6000 to >7FFF) simply by installing a double throw switch from pin 12 (or 13) of the 74LS21 to pin 13 of U504 (>6000 page select). If pin 12 is linked to pin 14 as per instructions above, this link must be removed and the connection to pin 14 is wired as one of the switch positions (see diagram). This memory space must not be used by any other cartridge or device while this switch is active. It is advisable that this requirement be incorporated as part of the switching. Switch pin 12 of 74LS21 to +5 volts (pin 14 of 74LS21) to de-activate this function.

2) The DSR space (>4000 to >5FFF) can also be made available in static RAM by switching pin 12 (or 13 if already used) of 74LS21 to pin 12 of U504 (>4000 page select). Switch pin 12 of 74LS21 to +5 volts (pin 14 of 74LS21) to de-activate this function. The same precaution with respect to accessing more than one active memory applies as mentioned above.

If all of these modifications are carried out, then you have effectively used 49K bytes out of the 64K byte upgrade. The remaining 15K bytes cannot be easily used at this time. Perhaps what we need now is some sort of indicator to show that all is working as intended?

Watch this space!

The next hardware project will describe how to produce a truly versatile "SuperModule" containing Minimem, Editor Assembler and 4 manually switched 8K byte banks of battery backed RAM at >6000.



L.A. MADRID, TISHUG
19/10/88

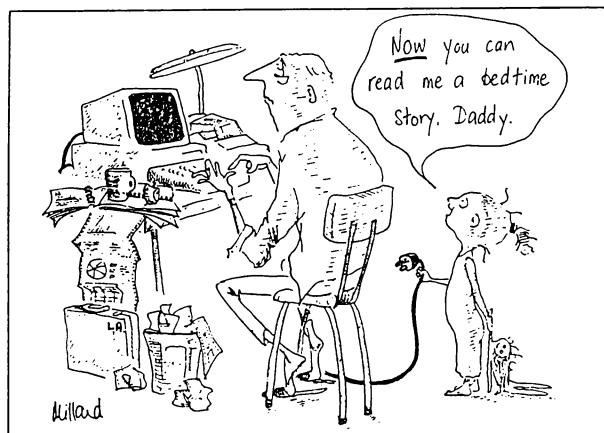
For Sale

Original CorComp Double Density Disk controller card with MG upgrade ROM. Also included in sale is TI-IBM connection module. Asking \$250. Peter Schubert, phone (02)358 5602 or write PO Box 28, Kings Cross, 2011.

TI99/4A Computer, cassette recorder and joysticks, Extended BASIC, Adventure Module, Touch Typing and many other game cartridges. Good Condition. \$160 o.n.o. Ring Dick on 449-1773, after 6pm.

Change of address

MAGNETIC DATA STORAGE has moved from Kensington to Summer Hill. The new address is 5 Grosvenor St, Summer Hill, phone (02)798 3833. This company sells MPI drives and is very capable of repairing disk drives.



Tips from Great Britain

by Stephen Shaw, UK

Greetings once more. Your letters of enquiry and requests are always welcome, a SAE for a direct reply please!:

10 Alstone Road, STOCKPORT, Cheshire, SK4 5AH

This is also the address for the disk library, with copying costs at just one pound per disk (plus one pound per order to cover post, packing etc etc)- why not send for a list of disk library contents? It comes on two SSD disks, and all you need to do is send two disks plus return post and packing.

I have been occupied recently grappling with the likes of MS-DOS, and a few other IBM "goodies" such as a data base with the capacity to handle records (records!) of up to 4K. It is a pleasure to get home to a well designed modern computer, and I do mean that!

First an item for Extended BASIC users with 32K RAM. One of the things Myarc intended to include in their XB was a forced garbage collection, but they seem to have run out of room. It is really quite easy, but what am I talking about?

Type in this program and run it:

```
100 CALL CLEAR
110 A$="1234567890"
120 FOR T=1 TO 12
130 CALL SPRITE(#T,42,2,30+3*T,30+3*T,0,34+T)
140 NEXT T
150 CALL PEEK(-31890,I,M)
160 CALL PEEK(-31974,H,L)
170 A$=A$&A$&A$&A$
180 DISPLAY AT(20,15):"FREE:"
190 F=(256*H+L)-(256*I+M)
200 DISPLAY AT(21,15):F
210 GOTO 140
220 END
```

The number you see counting down is the free stack space. It is being filled with redundant definitions of A\$ (line 170). When the number reaches or passes zero, you will see that it starts again from a high value, but at the moment of changeover, all your sprites will halt briefly. During this brief halt, the computer is checking the stack to see what information there is redundant (no longer required) and getting rid of anything it does not need, garbage collection.

Garbage collection not only causes sprites to halt but can also interfere with music, causing the odd note to sound for longer. Instead of having garbage collection occur when the stack is full, which can be quite often if the program is long and stack space short, it can be useful to force a garbage collection at a time we choose, when the effect may not be too obvious. Also, if we do a garbage collection before the stack is full, the pause may be quite a bit shorter.

Here is how to do it. Insert the following lines into the above program:

```
95 CALL INIT
205 CALL LOAD(-31885,144,"",-31858,81,169,152,0)
```

Now run the program again. Notice that the stack space remains constant, as we are forcing a garbage collection after each definition of A\$. How is sprite speed affected?

Try a garbage collection at an intermediate stage:

```
204 IF F>10000 THEN 210
```

Any difference? Have fun. Thanks to the Sydney User Group (TISHUG) in Australia for this one, taken from their newsletters for March and July 1987.

LOGO

At last a LOGO query, and a very interesting one too. The LOGO manual contains an ANIMAL program. If you key it in and build up the KNOWLEDGE a little, say by adding three or four extra choices, as the program

runs, then save it to disk, you will find that RECALLing the procedure will lock your console up. Hmmm.

On Page 7 of the Logo manual, you can find the information that input lines are limited to 127 characters.

Here is the clue! BASIC program lines are also limited to 127, 127 bytes after tokenisation has occurred, which is why you can sometimes key in less than the 5 screen lines Extended BASIC allows you and still get a LINE TOO LONG error message.

Back to LOGO. As you play ANIMAL, the KNOWLEDGE LIST gets longer and longer and longer and once it exceeds 127 bytes, it becomes incapable of loading back into the console. The file is just too long for the routine which reads the disk and places the definitions where they belong. Once you have been playing ANIMAL for a while, if you wish to save it, you must first reduce the LIST attached to KNOWLEDGE to under 127 bytes. You could do this in immediate mode or write a routine to do it- perhaps call it "RESET".

TURBO PASCAL 99

In the last issue I mentioned Texaments as a supplier of Turbo Pasc 99, a superb language for the TI99/4A, copyright June and August 1986 by an Austrian software house, with rights now owned by a German company.

Texaments advise that they no longer wish to offer this program, and as a result of the time and money I have wasted on this, I now delete Texaments from my accredited list.

Yes, a superb language, now 22 months old, and with no distribution in the USA, Canada, the UK, Australia, France. Available only to a small (very small) fraction of TI99/4A users.

If you have an interest in the language, please write to me.

MUSIC MAKER

I have received several requests on how to use Music Maker Module with a disk system, and as the original Music Maker had no disk capacity at all, I assumed that this was the case with all. But not so. We have yet another module appearing in two different guises.

Some Music Maker modules apparently let you save to disk, but then do not let you load back in, the simple reason being that Music Maker saves a FIXED LENGTH data file and that length of data file requires you to use CALL FILES(0) to load it back in. And you cannot use CALL FILES(0) can you!

MICROpendium December 1987 gave a complex solution, requiring no less than 14 complex steps, which change the length of the saved file from the former fixed 63 sectors (Display Fixed 128) to 59 sectors, which apparently you can then load in.

One of our members however has a module which saves in the required 59 sectors, but then will only load in 43 sectors.

These are differences of 1K and 4K respectively, and sound like nothing more than sloppy programming by TI - and suggest at least three versions of the module!!!!

You could use a sector editor to change the file length, taking care to see if the data is stored at the start of the file (no problem) or the end of the file (erase early sectors and move header forward, ugh).

It would be very nice if, now you know what is needed, one of our talented machine code programmers were to write a routine to input the original file name, read the disk, and make all the necessary changes for you. I cannot do it! Consider yourself challenged!

DISK DRIVES

SECTORS/TRACKS/DENSITIES/SIDES...aaaagh!

To allay a little confusion:

TI Standalone disk controller:

Single Sided, 40 track, Single Density, 360 sectors, 90K, 9 sectors/track

TI PEB Card: above or

Double Sided, 40 track, single density, 720 sectors, 180K, 9 sectors/track

Myarc/Corcomp PEB cards: above or
 Double Sided, 40 track, double density, 1440
 sectors, 360K, 18 sectors/track or
 1280 sectors, 320K, 16 sectors/track.
 Myarc optional extension to Myarc PEB Card:
 Double sided 80 track quad density (no other data to
 hand).

COMPATABILITY:

A disk recorded on a system higher in the list can
 be read on a system lower in the list (except 80
 track).

A disk recorded on a system lower in the list may
 not be read on a system higher in the list.

Except that a single sided system may be able to
 read some files on a double sided disk.

The Corcomp and Myarc controllers can both read
 and write in single or double density, so you can still
 record SSSD on these!

Disk Drives: All systems should be capable of
 working with any 40 track "standard" disk drive. An 80
 track drive is of value only if you have the extended
 Myarc controller card.

You may need to set small switches in the drives
 to match the controllers, if disk access time is
 variable. The TI Controller does not take advantage of
 faster access times whereas the CorComp and Myarc
 controllers can. Myarc can use an access time as fast
 as 6ms. There is no incompatibility unless the disk
 drive is too slow, which should be unlikely these
 days! All controllers can work with the same maximum
 access time as the original TI controller, which is
 much slower than any modern drive. The TI disk
 controller has a "hardware" fixed speed. The other
 controllers can be made to function more efficiently
 with the faster modern drives.

A letter from a member dated December 1987 advises
 of the partial delivery of an order sent to PILGRIMS
 PRIDE in October 1986. Speedy!

MYARC RAM CARD:

Why can you only use 400K of the Myarc
 RAM card for disk emulation? Because the disk usage is
 bit mapped into an area of the disk which can only map
 400K. There is no room to map more. If you want a
 bigger disk you must use a different mapping system,
 either increase the area in which the bit map is held,
 or allow a "floating" bit map on an "as required"
 basis. That would mean you would need a new disk
 controller, your RAM card would not be strictly
 compatible with your old software! Myarc were right to
 stick to 400K. The remainder is available for print
 spooling, and for your operating system to use as part
 of the normal computer memory map area.

FORTTRAN from Tenex

George Michel reports as follows: "I
 have the LGMA FORTRAN discs and I have succeeded in
 producing working object programs.... I concluded that
 the LGMA discs apply to a very powerful extract of the
 ANSI 77 full language, and is not to be confused with
 the ANSI 77 subset.

"The beauty and speed of the TI99/4A is brought
 out by the FORTRAN program. There are of course
 limitations such as: having to write a lot of
 subroutines because the compiler cannot handle large
 programs; requiring a deep pocket to stock up with
 printer paper; having sufficient interest to find out
 how the IBM mathematicians brains work, etc. It is
 perhaps worth mentioning that assembly language
 subroutines may be called on as well as FORTRAN
 subroutines, thus widening the applications field
 considerably." Thanks George.

Extended BASIC Programming Tip

Courtesy John Seager

Using ACCEPT AT for a numeric variable,
 how do you idiot proof it so the program will not
 bomb? You can insert a default input value, and use a
 negative size, and also use VALIDATE.

However, the user CAN blank the default variable
 with CLEAR, and if the input variable is a numeric
 variable, trying to input a blank will cause an error
 condition.

I have frequently seen authors input all numbers
 into a string variable. Inputting a blank does not
 cause an error, and you can test for a null input and
 go back if required.

You do not have to do it that way. John points
 out that using ON WARNING NEXT will test for the null
 input and go back for you. Try it. Eg.

```
100 ON WARNING NEXT
110 ACCEPT AT(4,5)ERASE ALL VALIDATE(DIGIT):A
120 GOTO 110
```

John also mentions that if you program:

```
100 ON ERROR 600
110 RUN "DSK1.NOFILE"
600 ON ERROR 600 :: RETURN
the RETURN will fail as the failed RUN seems to
remove the internal pointers.
```

This is a deliberate ploy by TI to avoid the
 "accidental" removal of the List Protection flag, which
 happens with Version 100 of Extended BASIC.

You need to use the format RETURN XXXX where XXXX
 is a line number to go to which will RUN your original
 program again. You could use RUN 110 or something if
 required. The second RUN will of course reset all
 required pointers.

=====

If you would like to add PEEKV and POKEV to your
 Extended Basic programming, the following program will
 help you. It comes from the April 1984 issue of The
 Smart Programmer and was written by John Brown:

```
100 CALL CLEAR :: CALL INIT :: CALL LOAD(8196,63,232)
110 CALL LOAD(16360,80,79,75,69,82,32, 38,12,80,79,75,
69,86,32,37,164,80,69,69,75,86,32,37,36)
120 CALL LOAD(9491,100)
130 CALL LOAD(9508,2,224,37,20,3,0,0,0, 2,0,0,100,200,
0,37,18,4,192,2,1,0,1,4,32,32,12,4,32)
140 CALL LOAD(9536,32,24,18,184,192,32, 131,74,2,1,37,
0,208,160,131,18, 9,130,2,34,255,255,4,32,32,44)
150 CALL LOAD(9562,4,197,209,34,36,255, 9,132,19,21,4,
195,60,224,37,18, 200,5,131,76,20,5,131,78,200,5)
160 CALL LOAD(-9588,131,80,2,5,64,0, 161,68,2,131,0,1,
17,6,2,5,65, 0,161,67,6,196,200,4,131,76)
170 CALL LOAD(9614,200,5,131,74,4,192, 192,66,5,129,4,
32,32,8,6,2,22,221,4,96,37,254)
180 CALL LOAD(9636,2,224,37,20,3,0,0,0, 4,192,2,1,0,1,
200,1,37,18,4,32, 32,12,4,32,32,24,18,184)
190 CALL LOAD(9664,200,32,121,74,37,0, 184,32,131,18,
37,19,2,3,0,2)
200 CALL LOAD(9680,4,192,192,67,4,32, 32,12,4,32,32,
24,18,184,216,224, 131,75,37,0,5,131,136,3)
210 CALL LOAD(9704,37,18,22,242,192,32, 37,0,2,1,37,2,
192,131,2,34,255,254,4,32,32,36)
220 CALL LOAD(9726,4,192,216,0,131,124, 2,224,131,224,
4,96,0,112)
230 CALL LOAD(9740,3,0,0,0,4,192,2,1,0, 1,4,32,32,12,
200,32,131,74,37, 18,2,1,0,2,4,32,32,12,4,32)
240 CALL LOAD(9770,32,24,18,184,192,32, 131,74,208,32,
37,19,4,32,32,48,4,91)
250 CALL LOAD(8194,39,104)
260 END
```

Now, type that in, RUN it, and you can now use in
 your programs:

```
CALL LINK("PEEKV",address,variable,variable,...)
CALL LINK("POKEV",address,value,value,value,...)
ADDRESS must be a VDP RAM address in DECIMAL (0 to
16383)
```

You may use up to 15 (fifteen) values or
 variables.

You also have:

```
CALL LINK("POKER",vdpregisterno,value)
to write a value to one of the VDP registers.
```

Using Peekv and Pokev and VDP registers has been
 covered in some depth in previous issues of TI*MES.
 Let me know if you want reprints!

=====

Games Information

by Robert Brown and Stephen Judd

Welcome to yet another Games Info. We have had a break for a while, due to many reasons, but we are now back again. I would like to say a big hello to all our readers from the SNUG User Group in America.

If you are planning to buy some games from the States, I recommend you do not buy Strike Three, a new baseball cartridge. Seeing both the cartridge and Baseball (an Extended BASIC program by Mario Beaulieu), the Extended BASIC program has better features, while the speed, sound and graphics are of the same standard as the cartridge.

Also available in America, is Karate Challenge. This sells for \$14.95 and is available from Triton. I have played this, and it is an excellent game, with 3 punches and 3 kicks available. A review will be coming soon.

Well, as I promised on the BBS, this month we have a review on Spad XIII.

Spad XIII is a machine language program run through Extended BASIC. Once loaded however, you will receive a title screen with a option or two.

Once started you are crushed with the problem of taking off, and to use the comment of a fluffy sock puppet on TV, "No problem".

Planes have three flight dimensions. Yaw is the rotation around the vertical axis of the plane, left and right. Pitch is the rotation moving the nose up and down. Roll is the rotation moving the wings up and down. If you did not understand that do not worry, nobody else did. Maybe here is a new competition?

When the stick is moved right the left aileron goes down and right goes up creating a right turn. If the stick is turned left the opposite occurs. The elevator is controlled by pushing the stick forward or backward causing a drop or rise in altitude, or gain or loss of speed. The best way to purposely stall is to pull back until the plane's speed is less than its stated stalling speed.

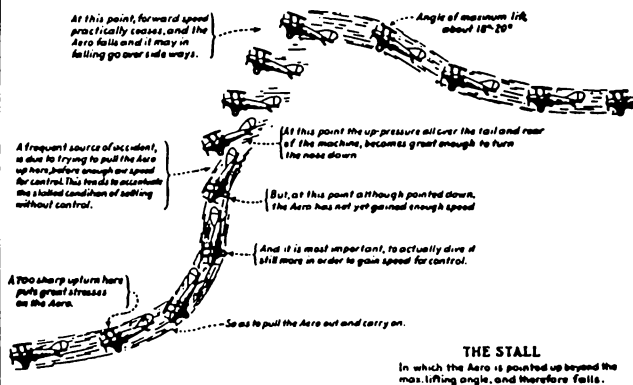


Figure 1 The Stall

Spad XIII is controlled either entirely by keyboard or part joystick and keyboard with joystick substituting for the stick and fire buttons. Two commands maybe input at one time eg, one from left keyboard (joystick) and one from right.

The keys E-S-D-X-W-R-Z-C are the plane's stick or joystick. Also A is strong left, F strong right and V strong back stick for acrobatic moves (the joystick has no corresponding positions to accomplish these maneuvers). The stick on screen will correspond to your action. Keys 1-2-3-4-5-6 are used as the views. The U key gives an unobstructed view of the front view and is restored by pressing a view key. Firing is only possible at the front. If a view is pressed constantly it will give a peek through the plane's wings and fuselage. When you look up (view 5), forward is down on the screen. When you look down (view 6), forward is up. Picture the pilot actually looking up or down forward is up or down in his seat.

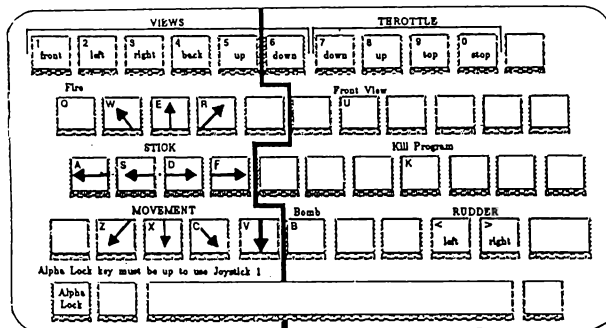


Figure 2 Keyboard Template

The throttle keys are 7-8-9-0 with 7 and 8 raise and lower the throttle in increments. 9 gives full throttle, and 0 stops the engine. (Even at 0, the air pressure may keep the propeller moving and cause the engine to run.)

Rudder keys are "<" and ">."

Fire key is Q or joystick #1 button. Bombs are dropped using B, which can be seen looking down and is seen exploding less than 1000 feet.

To finish press K, only when you are safely on the ground. it will display the final score and end the game.

At any time, you can quit by pressing FCTN[+] (Quit).

In Spad XIII, the screen represents the view of the pilot sitting in the cockpit. In addition to the wings, struts, and guns of the plane, the view includes an instrument panel and the outside scenery. The instrument panel consists of four gauges and two mechanical devices.

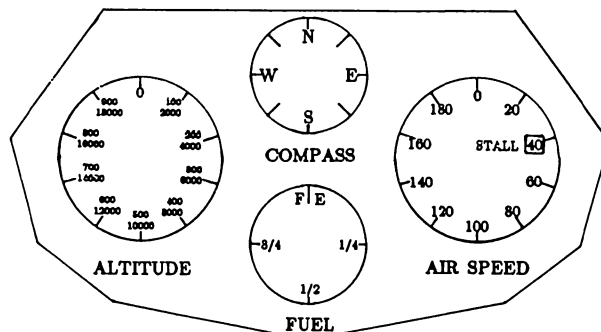


Figure 3 The Instrument Panel

The altitude gauge has two hands, one black and one white. The black hand indicates thousands of feet and the white hand indicates hundreds.

The air speed gauge has one hand and registers from 0 to 200. If your airspeed drops below 40 knots, a stall is likely.

The compass on the top is a standard magnetic device corrected for the mass of the engine.

The fuel gauge reads full at about 10 o'clock and empty when pointing straight up.

The throttle is the hand below and to the left of the instrument panel. Top is 1200 RPM. It is pegged at 200 RPM although the throttle changes in increments of 100 RPM.

We continue the review next month, where we look at the objects ground, your first flight, acrobatics and the purpose of the game. From the above information, you should be able to at least fly the plane.

100 REM (c) W.A.McMATH 1984

```

110 CALL CLEAR
120 CALL SCREEN(8)
130 RANDOMIZE
140 RESTORE 4000
150 READ BALL
160 GOTO 450
170 CALL HCHAR(22,5,32,96)
180 FOR I=1 TO LEN(MSG$)
190 CALL HCHAR(RN,4+I,ASC(SE
G$(MSG$,I,1)))
200 NEXT I
210 RETURN
220 MSG$="BLACK-"&STR$(RSC)&
" BLUE-"&STR$(BSC)&" ENDS-
"&STR$(NE)
230 RN=1
240 GOSUB 180
250 RETURN
260 K=K+1
270 A(K,1)=BALL
280 A(K,2)=INT((FIN-I)*RND)+
1
290 P=INT((BALL-126)/2)
300 F(P,1)=Y
310 F(P,2)=X
320 IF (CHK=144)*(FIN>10)THE
N 360
330 IF F(P,3)=1 THEN 350
340 F(P,3)=0
350 RETURN
360 F(P,3)=1
370 RETURN
380 MSG$="PRESS ANY KEY TO C
ONTINUE"
390 RN=24
400 GOSUB 180
410 CALL SOUND(100,660,0)
420 CALL KEY(3,P,Q)
430 IF Q=0 THEN 420
440 RETURN
450 B1$="003C427E7E7E3C"
460 B2$="003C7E7E7E423C"
470 CALL CHAR(104,B1$)
480 CALL CHAR(105,B2$)
490 CALL CHAR(112,"FFFFFFFF
FFFFFFFF")
500 CALL CHAR(120,"1FOFO70B1
1204080")
510 CALL CHAR(121,"080COEFFO
EOC08")
520 CALL CHAR(122,"804020110
BO70F1F")
530 FOR I=1 TO 4
540 CALL CHAR(126+I*2,B1$)
550 CALL CHAR(127+I*2,B2$)
560 CALL CHAR(134+I*2,B1$)
570 CALL CHAR(135+I*2,B2$)
580 NEXT I
590 CALL CHAR(152,"FFFFFFFF
FFFFFFFF")
600 CALL CHAR(144,"0000183C3
C18")
610 CALL COLOR(11,3,1)
620 CALL COLOR(10,2,8)
630 CALL COLOR(12,2,1)
640 CALL COLOR(13,2,3)
650 CALL COLOR(14,5,3)
660 CALL COLOR(15,16,3)
670 CALL COLOR(16,12,1)
680 PRINT "##### BOWLS
#####"
690 PRINT "THE NORMA
L RULES FOR THE"
700 PRINT "THE OBJECT OF THE
GAME IS"
710 PRINT "TO GET YOUR BOWLS
NEARER"
720 PRINT "TO THE JACK(THE SM
ALL WHITE"
730 PRINT "BALL) THAN YOUR O
PPONENT"
740 PRINT "EACH BOWL HAS
A BIAS WHICH"
750 PRINT "MAKES THE BOW
L TURN AND YOU"

```

```

760 PRINT "HAVE FOUR BOWLS E
ACH END"
770 PRINT "IF YOUR BOWL G
OES IN THE "
780 PRINT "DITCH IT IS LO
ST,UNLESS"
790 PRINT "IT HIT THE JACK,I
N WHICH"
800 PRINT "CASE IT STAYS IN P
LAY"
810 PRINT "PRESS ANY KEY TO C
ONTINUE"
820 GOSUB 420
830 CALL CLEAR
840 PRINT "ONE SET OF BOWLS
ARE BLUE"
850 PRINT "THE OTHER SET BLA
CK"
860 PRINT "THE BOWLS ARE DELIVERE
D"
870 PRINT "ALTERNATELY(BLACK
,BLUE ETC.)"
880 PRINT "EACH PLAYER TAKES
A TURN AT"
890 PRINT "DELIVERING THE
FIRST BOWL"
900 PRINT "AT ALTERNATE EN
DS"
910 PRINT "THE BIAS OF THE B
OWL IS"
920 PRINT "MARKED WITH A LINE
AND THE"
930 PRINT "BOWL BENDS TOWARDS
THE BIAS"
940 PRINT "FOR EACH BOWL YOU
MUST"
950 PRINT "CHOOSE YOUR DELIVERY"
960 PRINT "POSITION,DIRECTION,BIAS A
ND"
970 PRINT "STRENGTH"
980 PRINT "THE STRENGTH OF T
HE BOWL"
990 PRINT "DETERMINES HOW FAR
IT"
1000 PRINT "TRAVELS AND WHEN THE
BIAS"
1010 PRINT "STARTS TO CHANGE THE"
1020 PRINT "DIRECTION"
1030 PRINT "PRES
S ANY KEY TO CONTINUE"
1040 GOSUB 420
1050 CALL CLEAR
1060 PRINT "YOU CAN CHOOSE TO
PLAY"
1070 PRINT "A ONE OR TWO PLAYER
GAME"
1080 PRINT "IN THE ONE PLAYER
GAME YOU"
1090 PRINT "PLAY THE BLACK B
OWLS AND"
1100 PRINT "THE COMPUTER PLAY
S THE BLUE"
1110 PRINT "THE SCORE AND THE
NUMBER"
1120 PRINT "OF ENDS PLAYED ARE
SHOWN AT"
1130 PRINT "THE TOP OF THE S
CREEN"
1140 PRINT "THE COMPUTER WILL
MAKE THIS"
1150 PRINT "SOUND WHEN YOU
HAVE TO"
1160 PRINT "DO SOMETHING"
1170 PRINT "PRESS ANY KEY"
1180 GOSUB 420
1190 CALL SOUND(100,660,0)
1200 CALL CLEAR
1210 PRINT "WHICH GAME DO YOU
WISH"
1220 PRINT "1--ONE PLAYER
GAME"
1230 PRINT "2--TWO PLAYER
GAME"
1240 PRINT "
(PRESS 1 OR
2)"
1250 CALL KEY(3,P,Q)
1260 IF (Q=0)+(P<49)+(P>50)TH
EN 940
1270 NP=P-49
1280 CALL CLEAR
1290 PRINT "WOULD YOU LIKE TO
PLAY"
1300 PRINT "1-A LIMITED NUMBE
R OF ENDS"
1310 PRINT "2-TO A WINNING T
OTAL"
1320 PRINT "
(PRESS 1 OR
2)"
1330 CALL KEY(3,P,Q)
1340 IF (Q=0)+(P<49)+(P>50)T
HEN 1010
1350 IF P=49 THEN 1080
1360 PRINT "TYPE IN THE NUMB
ER OF SHOTS"
1370 PRINT "
(THEN PRESS
ENTER KEY)"
1380 INPUT LV
1390 TNE=1000
1400 GOTO 1110
1410 PRINT "TYPE IN THE NUMB
ER OF ENDS"
1420 PRINT "
(THEN PRESS
ENTER KEY)"

```

```

1430 INPUT TNE
1440 LV=1000
1450 NE=0
1460 SH=0
1470 BSC=0
1480 RSC=0
1490 XD=1
1500 CALL CLEAR
1510 K=0
1520 FOR I=1 TO 4
1530 F(I,3)=0
1540 F(I+4,3)=0
1550 C(I)=100
1560 D(I)=100
1570 NEXT I
1580 FOR I=2 TO 20
1590 CALL HCHAR(I,5,112,25)
1600 NEXT I
1610 CALL HCHAR(2,5,152,25)
1620 CALL HCHAR(20,5,152,25)
1630 CALL VCHAR(2,29,152,19)
1640 CALL VCHAR(2,30,152,19)
1650 FOR I=1 TO 6
1660 CALL HCHAR(4+I*2,5,48+I
)
1670 NEXT I
1680 PY=11
1690 PX=20+INT(RND*7)
1700 CALL HCHAR(PY,PX,144)
1710 READ BALL
1720 IF BALL>-1 THEN 1410
1730 RESTORE 4000
1740 GOTO 1370
1750 IF BALL=0 THEN 3090
1760 IF BALL>135 THEN 1470
1770 MSG$="BLACK TO PLAY (NO
.&STR$((BALL-126)/2)&)"
1780 RN=21
1790 GOSUB 170
1800 GOTO 1680
1810 MSG$="BLUE TO PLAY (NO.
.&STR$((BALL-134)/2)&)"
1820 RN=21
1830 GOSUB 170
1840 IF NP>0 THEN 1680
1850 V=INT((PX-14)/2)+INT(RN
D*2)
1860 IF V<8 THEN 1540
1870 V=7
1880 Q=2-(V>3)-(V>5)
1890 P=SGN(RND-.5)
1900 Y=PY+P*Q
1910 IF (Y>6)*(Y<16)THEN 159
0
1920 Y=PY-P*Q
1930 IF Y<PY THEN 1620
1940 B=0
1950 GOTO 1630
1960 B=1
1970 YD=0
1980 X=6
1990 BALL=BALL+B
2000 FIN=9+INT((V*1.8))
2010 GOTO 2200
2020 MSG$="POSITION(1-6)"
2030 RN=22
2040 GOSUB 170
2050 CALL SOUND(100,660,0)
2060 CALL KEY(3,P,Q)
2070 IF (Q=0)+(P<49)+(P>54)T
HEN 1720
2080 Y=4+(P-48)*2
2090 X=6
2100 MSG$="DIRECTION(1-3)"
2110 RN=22
2120 GOSUB 170
2130 CALL HCHAR(24,8,120)
2140 CALL HCHAR(24,10,49)
2150 CALL HCHAR(24,14,121)
2160 CALL HCHAR(24,16,50)
2170 CALL HCHAR(24,20,122)
2180 CALL HCHAR(24,22,51)
2190 CALL SOUND(100,660,0)
2200 CALL KEY(3,P,Q)

```

continued on page 17

```

10 REM RUNS IN TIB OR EX BAS
30 DEF RAN(X)=INT(X*RND+1)
40 AC$="10410510610710810911
0111112113116117066067068069
070071072073074075076077"
50 GOSUB 7000
60 CALL CLEAR
70 DIM R1(32),R2(32),V1(32),
V2(32)
100 RANDOMIZE
110 REM FOLLOWING TWO LINES
MUST* BE RETAINED
120 PRINT "BLACK BOX": : "THI
S PROGRAMME IS BASED UPON TH
E GAME 'BLACK BOX':"PUBLIS
ED BY WADDINGTONS"
130 PRINT "HOUSE OF GAMES
LEEDS": "& INVENTED BY": "DR.E
RIC SOLOMON": : "ALL RIGHTS R
ESERVED": "USED BY PERMISSION
"
139 RESTORE 180
140 FOR I=96 TO 121
150 READ A$
160 CALL CHAR(I,A$)
170 NEXT I
171 CALL HCHAR(4,28,120)
172 CALL HCHAR(5,28,121)
180 DATA 000000FF,101010FF10
101010,1010101010101010,1824
24247E428181,784444447848444
2,0,0,0
190 DATA FF818199998181FF,FF
9999999999FF,EE888888EE222
2EE
200 DATA 187E5A7E3C2424E7,FF
C3A59999A5C3FF,1824429999422
418
210 DATA FF81FF81FF8181FF,FF
9999FFFF9999FF,FF99FF8181424
224,E7848484848484E7,000000F
F,1010101010101010
220 DATA 00402010080402,0002
0408102040,0,0,FF80BE8888B88
OFF,017D417D057D01FF
230 REM LAST TWO=LOGO SJS
260 PRINT "DO YOU WANT INSTR
UCTIONS?": "PRESS Y OR N": :
270 CALL KEY(0,A,VR)
280 IF VR<1 THEN 270
290 IF A<>89 THEN 300
292 PRINT " ": : "PLEASE WA
IT": : :
294 GOTO 4000
300 IF A<>78 THEN 270
310 CALL CLEAR
320 GOSUB 6000
330 M$="HOW MANY ATOMS?(<10)
"
340 R=23
350 VR=3
360 GOSUB 3700
365 VR=24
370 GOSUB 3500
380 N=VAL(IN$)
390 IF (N<1)+(N>9) THEN 330
392 M$="ATOMS:"&STR$(N)
393 R=1
394 VR=3
395 GOSUB 3700
396 CALL HCHAR(23,1,32,32)
400 REM *** GAME CALCULATION

410 FOR J=0 TO 9
420 FOR I=0 TO 9
430 B(I,J)=0
440 NEXT I
450 NEXT J
460 FOR I=1 TO N
470 X=RAN(8)
480 Y=RAN(8)
490 IF B(X,Y)<>0 THEN 470
500 B(X,Y)=1
510 NEXT I
520 S=0

```

```

530 C=0
540 M$="RAY? "
550 R=1
560 VR=24
570 GOSUB 3700
575 VR=28
580 GOSUB 3500
590 RY=VAL(IN$)
595 IF RY=0 THEN 1490
600 IF (RY<1)+(RY>32) THEN 54
0
610 ON INT((RY-1)/8)+1 GOTO
620,670,720,770
620 X=0
630 Y=RY
640 U=1
650 V=0
660 GOTO 810
670 X=RY-8
680 Y=9
690 U=0
700 V=-1
710 GOTO 810
720 X=9
730 Y=25-RY
740 U=-1
750 V=0
760 GOTO 810
770 X=33-RY
780 Y=0
790 U=0
800 V=1
810 X1=X+U
820 Y1=Y+V
830 IF U<>0 THEN 890
840 X2=X1-1
850 X3=X1+1
860 Y2=Y1
870 Y3=Y1
880 GOTO 930
890 Y2=Y1-1
900 Y3=Y1+1
910 X2=X1
920 X3=X1
930 ON 8*B(X1,Y1)+B(X2,Y2)+2
*B(X3,Y3)+1 GOTO 1030,1060,1
080,1060,940,940,940,940,940
,940,940,940,940,940,940,940
940 M$="ABSORBED"
950 R=2
960 VR=24
965 CALL HCHAR(R1(RY),V1(RY)
,32)
966 CALL HCHAR(R2(RY),V2(RY)
,99)
970 GOSUB 3700
980 S=S+1
990 FOR XX=1 TO 600
1000 NEXT XX
1010 CALL HCHAR(2,24,32,8)
1020 GOTO 540
1030 X=X1
1040 Y=Y1
1050 GOTO 1150
1060 Z=1
1070 GOTO 1090
1080 Z=-1
1090 IF U<>0 THEN 1130
1100 U=Z
1110 V=0
1120 GOTO 1150
1130 U=0
1140 V=Z
1150 ON INT((X+15)/8) GOTO 11
90,1170,1210
1160 STOP
1170 ON INT((Y+15)/8) GOTO 12
30,810,1250
1180 STOP
1190 Z=Y
1200 GOTO 1260
1210 Z=25-Y
1220 GOTO 1260
1230 Z=33-X
1240 GOTO 1260

```

```

1250 Z=8+X
1260 IF Z<>RY THEN 1360
1270 M$="REFLECTED"
1275 CALL HCHAR(R1(RY),V1(RY)
,32)
1276 CALL HCHAR(R2(RY),V2(RY)
,100)
1280 R=2
1290 VR=24
1300 GOSUB 3700
1310 FOR XX=1 TO 800
1320 NEXT XX
1330 S=S+1
1340 CALL HCHAR(2,24,32,9)
1350 GOTO 540
1360 M$="TO "&STR$(Z)
1370 R=2
1380 VR=24
1390 GOSUB 3700
1391 FOR XX=1 TO 700
1392 NEXT XX
1400 S=S+2
1410 R=VAL(SEG$(AC$,1,3))
1420 CALL HCHAR(R1(RY),V1(RY)
,32)
1430 CALL HCHAR(R2(RY),V2(RY)
,R)
1440 CALL HCHAR(R1(Z),V1(Z),
32)
1450 CALL HCHAR(R2(Z),V2(Z),
R)
1460 AC$=SEG$(AC$,4,LEN(AC$)
-3)
1470 CALL HCHAR(2,24,32,8)
1480 GOTO 540
1490 FOR R=1 TO 32
1500 CALL GCHAR(R2(R),V2(R),
VR)
1510 CALL HCHAR(R1(R),V1(R),
32)
1520 IF R<9 THEN 1570
1525 IF (R<25)*(VR<60) THEN 1
560
1530 IF R<25 THEN 1570
1535 IF VR>60 THEN 1570
1540 CALL HCHAR(R2(R),V2(R),
81-R)
1545 GOTO 1570
1550 IF R<9 THEN 1570
1560 CALL HCHAR(R2(R),V2(R),
32)
1570 NEXT R
1580 M$=" WHERE"
1590 R=1
1600 VR=24
1610 GOSUB 3700
1620 M$="ARE THE"
1630 R=2
1640 VR=24
1650 GOSUB 3700
1660 M$="ATOMS?"
1670 R=3
1680 VR=25
1690 GOSUB 3700
1700 M$="ATOM "
1710 R=5
1720 VR=25
1730 GOSUB 3700
1740 M$="ROW: COL:"
1750 R=6
1760 VR=24
1770 GOSUB 3700
1780 FOR Q=1 TO N
1790 M$=STR$(Q)
1800 R=5
1810 VR=30
1820 GOSUB 3700
1830 R=Q+8
1840 VR=25
1850 GOSUB 3500
1855 CALL HCHAR(19,26,32,6)
1860 I=VAL(IN$)
1870 IF (I<1)+(I>8) THEN 1830
1880 VR=30
1890 GOSUB 3500

```



```

1900 J=VAL(IN$)
1910 IF (J<1)+(J>8)THEN 1880
1920 IF B(I,J)<>1 THEN 1930
ELSE 1950
1930 S=S+5
1940 GOTO 1970
1950 B(J,I)=2
1960 C=C+1
1970 NEXT Q
1980 FOR J=1 TO 8
1990 FOR I=1 TO 8
2000 IF B(I,J)<1 THEN 2040
2010 R=2*I+3
2020 VR=2*I+4
2030 CALL HCHAR(R,VR,42)
2040 NEXT I
2050 NEXT J
2060 M$=STR$(C)&" OUT OF "&S
TR$(N)&" CORRECT"
2070 R=23
2080 VR=3
2090 GOSUB 3700
2100 M$="SCORE "&STR$(S)&" P
OINTS "
2110 R=1
2120 VR=3
2130 GOSUB 3700
2140 M$="ANOTHER"
2150 R=17
2160 VR=24
2170 GOSUB 3700
2180 M$="GAME?"
2190 R=18
2200 VR=24
2210 GOSUB 3700
2220 R=21
2230 VR=24
2240 M$="PRESS "
2250 GOSUB 3700
2260 M$="Y OR N "
2270 R=22
2280 VR=24
2290 GOSUB 3700
2300 CALL KEY(O,R,VR)
2310 IF VR<1 THEN 2300
2320 IF R=89 THEN 2350
2330 IF R<>78 THEN 2300
2340 STOP
2350 AC$="104105106107108109
110111121131161170660670680
69070071072073074075076077"
2360 GOTO 310
3500 IN$=""
3510 REM ** INPUT **
3520 CALL HCHAR(22,27,32,4)
3530 CALL KEY(O,M,ZN)
3540 CALL HCHAR(R,VR,30)
3550 CALL HCHAR(R,VR,32)
3560 IF ZN<1 THEN 3530
3570 IF M=13 THEN 3680
3580 IF (M<47)+(M>57)THEN 35
30
3590 CALL HCHAR(R,VR,M)
3600 VR=VR+1
3610 IF VR<33 THEN 3650
3620 VR=32
3630 M=13
3640 GOTO 3680
3650 IF CHR$(M)="" THEN 3510
3660 IN$=IN$&CHR$(M)
3670 GOTO 3510
3680 IF IN$="" THEN 3510
3682 COL=VR
3684 ROW=R
3686 M$="WAIT"
3688 R=19
3690 VR=26
3692 GOSUB 3700
3694 R=ROW
3696 VR=COL
3698 RETURN
3700 CALL HCHAR(19,26,32,6)
3710 FOR CT=1 TO LEN(M$)
3720 CALL HCHAR(R,VR,ASC(SEG
$(M$,CT,1)))

```

```

3730 VR=VR+1
3740 NEXT CT
3750 CALL SOUND(200,700,0)
3760 RETURN
4000 REM *INSTRUCTIONS & DEM
O
4010 CALL CLEAR
4020 PRINT "BLACK BOX": : :
:
4030 REM
4040 PRINT "THIS PROGRAMME B
ASED ON ONE": "BY JEFF KENTON
PUBLISHED IN": "CREATIVE COM
PUTING (C)1978"
4050 PRINT "SUB ENQUIRIES TO
": "HAZEL GORDON,27,ANDREW C
LOSE": "STOKE GOLDING,NUNEATO
N": "CV12 6EL U.K."
4060 PRINT "WRITTEN FOR 99/4
BY": "STEPHEN SHAW,10 ALSTON
E RD": "STOCKPORT,CHESHIRE,SK
4 5AH"
4070 PRINT " ": "PRESS ANY KE
Y TO CONTINUE"
4200 CALL HCHAR(1,28,120)
4210 CALL HCHAR(2,28,121)
4220 CALL KEY(O,R,VR)
4230 IF VR<1 THEN 4220
4240 CALL CLEAR
4250 CALL SCREEN(2)
4260 PRINT "YOU HAVE A BOX W
ITH 32": "WINDOWS. IT CONTAIN
S ATOMS": "YOU MUST LOCATE TH
EM BY"
4270 PRINT "SHINING RAYS INT
O THE": "WINDOWS": "THE COMPUT
ER WILL TELL YOU ": "IF THEY
ARE ABSORBED,"
4280 PRINT "REFLECTED,OR DEF
LECTED": "(& WHERE TO)": "A L
OW SCORE BEATS ": "A HIGH SCO
RE":
4290 PRINT "PRESS ANY KEY TO
CONTINUE"
4300 CALL SCREEN(4)
4310 CALL KEY(O,R,VR)
4320 IF VR<1 THEN 4310
4330 CALL CLEAR
4400 PRINT "THE NEXT DISPLAY
WILL": "ILLUSTRATE ATOMS (*)
AND": "RAYS. ": "REMEMBER:"
4410 PRINT "IN PLAY ATOMS AN
D RAYS": "ARE INVISIBLE!"
4420 PRINT "A=ABSORBED": "R=R
EFLECTED": "TO ENTER GUESSE
S,ENTER": " RAY=0 (ZERO)
"
4430 PRINT " ": "PRESS ANY K
EY TO CONTINUE"
4440 CALL HCHAR(1,28,120)
4450 CALL HCHAR(2,28,121)
4600 CALL KEY(O,R,VR)
4610 IF VR<1 THEN 4600
4620 CALL CLEAR
4630 GOSUB 6000
4640 RESTORE 4770
4650 FOR CT=1 TO 150
4660 READ R,VR,CODE
4670 IF R=1 THEN 4700
4680 CALL HCHAR(R,VR,CODE)
4690 NEXT CT
4700 R=23
4710 VR=3
4720 M$="PRESS ANY KEY TO CO
NTINUE"
4730 GOSUB 3700
4732 CALL KEY(O,R,VR)
4734 IF VR<1 THEN 4732
4736 CALL CLEAR
4750 RESTORE
4760 PRINT "BLACK BOX": : :
PLEASE WAIT": : : : : :
: : :
4765 GOTO 310

```

```

4770 DATA 5,10,115,7,10,116,
7,12,114,7,14,114
4780 DATA 7,16,117,5,16,115,
9,8,42,9,18,42,9,20,42
4790 DATA 15,12,42,15,14,42,
19,14,42,7,3,32,9,3,65,11,3,
32
4800 DATA 7,24,82,9,24,65,11
,24,82
4810 DATA 13,6,114,13,8,114,
13,10,117,11,10,117,11,12,11
4,11,14,114
4820 DATA 11,16,116,13,16,11
6,13,18,114,13,20,114
4830 DATA 17,6,114,17,8,114,
17,10,116,19,10,115
4840 DATA 17,20,114,17,18,11
4,17,16,114,17,24,82,1,1,32,
1,1,32
4850 STOP
4860 BREAK
6000 REM SET UP BOARD
6010 CALL CLEAR
6020 FOR R=4 TO 20 STEP 2
6030 FOR VR=6 TO 20 STEP 2
6040 CALL HCHAR(R,VR,96)
6050 CALL HCHAR(R,VR-1,97)
6055 IF R=4 THEN 6070
6060 CALL HCHAR(R-1,VR-1,98)
6070 NEXT VR
6080 NEXT R
6090 FOR R=4 TO 18 STEP 2
6100 CALL HCHAR(R,21,97)
6110 CALL HCHAR(R+1,21,98)
6120 NEXT R
6130 CALL HCHAR(20,21,97)
6140 FOR R=1 TO 32
6150 CODE=INT(R/10)+48
6160 IF CODE<>48 THEN 6180
6170 CODE=32
6180 CALL HCHAR(R1(R),V1(R),
CODE)
6190 CALL HCHAR(R2(R),V2(R),
R-10*INT(R/10)+48)
6200 NEXT R
6210 M$="PRESS ENTER AT END
OF ENTRY"
6220 R=24
6230 VR=3
6240 GOSUB 3700
6250 RETURN
7000 REM ARRAY R1 V1=TENS
7001 REM R2 V2 = UNITS
7004 PRINT " ": : : : : :
: : "PLEASE WAIT": : : : : :
: : : : : : :
7010 RESTORE 7050
7020 FOR CT=1 TO 32
7030 READ R1(CT),R2(CT),V1(C
T),V2(CT)
7040 NEXT CT
7050 DATA 5,5,3,4,7,7,3,4,9,
9,3,4,11,11,3,4,13,13,3,4,15
,15,3,4,17,17,3,4,19,19,3,4
7060 DATA 21,22,6,6,21,22,8,
8,21,22,10,10,21,22,12,12,21
,22,14,14,21,22,16,16,21,22,
18,18,21,22,20,20
7070 DATA 19,19,22,23,17,17,
22,23,15,15,22,23,13,13,22,2
3,11,11,22,23,9,9,22,23,7,7,
22,23,5,5,22,23
7080 DATA 2,3,20,20,2,3,18,1
8,2,3,16,16,2,3,14,14,2,3,12
,12,2,3,10,10,2,3,8,8,2,3,6,
6
7090 CALL SCREEN(4)
7100 CALL COLOR(9,3,1)
7110 CALL COLOR(10,7,1)
7120 CALL COLOR(11,16,1)
7130 CALL CLEAR
7140 RESTORE
7150 RETURN

```

continued from page 14

```

1870 IF (Q=0)+(P<49)+(P>51)T
HEN 1860
1880 YD=(P-48)-2
1890 MSG$="BIAS (1 OR 2)"
1900 RN=22
1910 GOSUB 170
1920 CALL HCHAR(24,8,104)
1930 CALL HCHAR(24,10,49)
1940 CALL HCHAR(24,14,105)
1950 CALL HCHAR(24,16,50)
1960 CALL SOUND(100,660,0)
1970 CALL KEY(3,P,Q)
1980 IF (Q=0)+(P<49)+(P>50)T
HEN 1970
1990 B=P-49
2000 BALL=BALL+B
2010 MSG$="STRENGTH(1-9)"
2020 RN=22
2030 GOSUB 170
2040 CALL SOUND(100,660,0)
2050 CALL KEY(3,P,Q)
2060 IF (Q=0)+(P<49)+(P>57)T
HEN 2050
2070 V=P-48
2080 FIN=9+INT((V*1.8)+RND*2
-ABS(YD)*3)
2090 CALL HCHAR(Y,X,BALL)
2100 MSG$="1-TO BOWL 2-FOR
CHANGES"
2110 RN=22
2120 GOSUB 170
2130 CALL SOUND(100,660,0)
2140 CALL KEY(3,P,Q)
2150 IF (Q=0)+(P<49)+(P>50)T
HEN 2140
2160 IF P=49 THEN 2200
2170 CALL HCHAR(Y,X,112)
2180 BALL=BALL-B
2190 GOTO 1680
2200 CALL HCHAR(21,5,32,96)
2210 FOR I=1 TO FIN
2220 IF (I<(9+V+RND*2-ABS(YD
)*4))+(K>0)THEN 2240
2230 YD=B+(B<1)
2240 CALL GCHAR(Y+YD,X+XD,CH
K)
2250 IF CHK=112 THEN 2380
2260 IF CHK=152 THEN 2620
2270 GOSUB 260
2280 CALL SOUND(-100,110,0,-
8,0)
2290 FIN=INT(FIN-I)+1
2300 BALL=CHK
2310 Y=Y+YD
2320 X=X+XD
2330 HW=F(INT((BALL-126)/2),
3)
2340 YD=YD+INT(RND*3)-1
2350 YD=YD-(YD<-1)
2360 YD=YD+(YD>1)
2370 GOTO 2210
2380 CALL HCHAR(Y,X,112)
2390 CALL HCHAR(Y+YD,X+XD,BA
LL)
2400 IF BALL=144 THEN 2460
2410 IF I<FIN THEN 2480
2420 P=INT((BALL-126)/2)
2430 F(P,1)=Y+YD
2440 F(P,2)=X+XD
2450 GOTO 2480
2460 PX=X+XD
2470 PY=Y+YD
2480 Y=Y+YD
2490 X=X+XD
2500 NEXT I
2510 HW=0
2520 IF K<1 THEN 1370
2530 BALL=A(K,1)
2540 FIN=A(K,2)
2550 P=INT((BALL-126)/2)
2560 Y=F(P,1)
2570 X=F(P,2)
2580 HW=F(P,3)
2590 YD=INT(RND*3)-1
2600 K=K-1

```

```

2610 GOTO 2210
2620 CALL HCHAR(Y,X,112)
2630 CALL HCHAR(Y+YD,X+XD,BALL)
2640 IF BALL=144 THEN 2940
2650 IF (HW>0)*(Y+YD>2)*(Y+YD<20) THEN 2740
2660 MSG$="IN THE DITCH-WASTED"
2670 RN=22
2680 GOSUB 170
2690 CALL HCHAR(Y+YD,X+XD,152)
2700 P=INT((BALL-126)/2)
2710 F(P,1)=0
2720 F(P,2)=0
2730 IF K>0 THEN 2530 ELSE 1370
2740 MSG$="IN THE DITCH-TOUCHER"
2750 RN=22
2760 GOSUB 170
2770 CALL HCHAR(Y+YD,X+XD,152)
2780 CALL GCHAR(Y+YD,30,CHK)
2790 IF CHK=152 THEN 2850
2800 IF Y>10 THEN 2830
2810 Y=Y+1
2820 GOTO 2780
2830 Y=Y-1
2840 GOTO 2780
2850 CALL HCHAR(Y+YD,30,BALL)
2860 P=INT((BALL-126)/2)
2870 F(P,1)=Y+YD
2880 F(P,2)=30
2890 HW=0
2900 IF BALL<>144 THEN 2930
2910 PX=30
2920 PY=Y+YD
2930 IF K>0 THEN 2530 ELSE 1370
2940 CALL HCHAR(Y+YD,X+XD,152)
2950 CALL GCHAR(Y+YD,30,CHK)
2960 IF CHK=152 THEN 3020
2970 IF Y>10 THEN 3000
2980 Y=Y+1
2990 GOTO 2950
3000 Y=Y-1
3010 GOTO 2950
3020 CALL HCHAR(Y+YD,30,144)
3030 PX=30
3040 PY=Y+YD
3050 MSG$="JACK IN DITCH"
3060 RN=22
3070 GOSUB 170
3080 IF K>0 THEN 2530 ELSE 1370
3090 MSG$="MEASURING FOR SHOTS"
3100 RN=22
3110 GOSUB 170
3120 NE=NE+1
3130 SH=1
3140 FOR I=1 TO 8
3150 IF (F(I,1)=0)*(F(I,2)=0) THEN 3200
3160 IF I>4 THEN 3190
3170 D(I)=SQR((PY-F(I,1))^2+(PX-F(I,2))^2)
3180 GOTO 3200
3190 C(I-4)=SQR((PY-F(I,1))^2+(PX-F(I,2))^2)
3200 NEXT I
3210 FOR I=1 TO 4
3220 IF D(1)<=D(I) THEN 3260
3230 CHK=D(1)
3240 D(1)=D(I)
3250 D(I)=CHK
3260 NEXT I
3270 FOR I=1 TO 4
3280 IF C(1)<=C(I) THEN 3320
3290 CHK=C(1)
3300 C(1)=C(I)

```

```

3310 C(I)=CHK
3320 NEXT I
3330 IF (C(1)=100)*(D(1)=100)
THEN 3730
3340 IF D(1)=C(1)THEN 3460
3350 IF D(1)<C(1)THEN 3410
3360 FOR I=2 TO 4
3370 IF C(I)>=D(1)THEN 3390
3380 SH=SH+1
3390 NEXT I
3400 GOTO 3470
3410 FOR I=2 TO 4
3420 IF D(I)>=C(1)THEN 3440
3430 SH=SH+1
3440 NEXT I
3450 GOTO 3600
3460 ON INT(RND*2)+1 GOTO 34
70,3600
3470 BSC=BSC+SH
3480 IF SH=1 THEN 3510
3490 B1$="SHOTS."
3500 GOTO 3520
3510 B1$="SHOT."
3520 MSG$="BLUE LIES "&STR$(
SH)&" "&B1$
3530 RN=22
3540 GOSUB 170
3550 GOSUB 220
3560 GOSUB 380
3570 IF BSC>=LV THEN 3880
3580 IF NE%=TNE THEN 3800
3590 GOTO 1170
3600 RSC=RSC+SH
3610 IF SH=1 THEN 3640
3620 B1$="SHOTS."
3630 GOTO 3650
3640 B1$="SHOT."
3650 MSG$="BLACK LIES "&STR$(
SH)&" "&B1$
3660 RN=22
3670 GOSUB 170
3680 GOSUB 220
3690 GOSUB 380
3700 IF RSC>=LV THEN 3910
3710 IF NE%=TNE THEN 3800
3720 GOTO 1170
3730 MSG$="NO COUNTING BOWL"
3740 RN=22
3750 GOSUB 170
3760 GOSUB 220
3770 GOSUB 380
3780 IF NE%=TNE THEN 3800
3790 GOTO 1170
3800 IF BSC>RSC THEN 3880
3810 IF BSC<RSC THEN 3910
3820 CALL CLEAR
3830 PRINT "THE SCORES ARE E
QUAL": ""; "PRESS 1 FOR DECIDI
NG END": ""; "PRESS 2 OTHERWIS
E":
3840 CALL KEY(3,P,Q)
3850 IF (Q=0)+(P<49)+(P>50)T
HEN 3840
3860 IF P=49 THEN 1160
3870 GOTO 3930
3880 CALL CLEAR
3890 PRINT "BLUE WON ::";BSC
; "-" ;RSC
3900 GOTO 3930
3910 CALL CLEAR
3920 PRINT "BLACK WON ::";RS
C; "-" ;BSC
3930 PRINT "#####
#####": "" ; ""
3940 PRINT "PRESS 1 TO PLAY
AGAIN"
3950 PRINT "PRESS 2 TO STOP"
3960 CALL KEY(3,P,Q)
3970 IF (Q=0)+(P<49)+(P>50)T
HEN 3960
3980 IF P=49 THEN 910
3990 END
4000 DATA 0,128,136,130,138,
132,140,134,142,0,136,128,13
8,130,140,132,142,134,-10

```

Jenny's Younger Set

Dear Jenny,

I have made some improvements to the program I sent you in August. The improvements in the August issue were for the program I sent you in July. Here are the improvements for the August program.

On line 450 the improvement is

```
450 H=INT(10*RND)+1 :: IF H>6 THEN 110
```

This is to prevent a number being divided by 0. Line 440 can also be altered.

```
440 G=INT(18*RND)+1
```

This is to prevent a "0/X". You would still get some hard ones such as 12/7 from time to time, but this cannot be helped. Maybe if lines 640 and 650 were changed as thus:

```
640 IF DF=INT(G/H) THEN RIGHT=RIGHT+1 ELSE
```

```
WRONG=WRONG+1
```

```
650 IF DF=INT(G/H) THEN PRINT "Well done ";NAME$ ELSE  
PRINT "Work on that division"
```

Line 460 can be changed as thus:

```
460 I=INT(10*RND) :: IF D>I+H THEN 110
```

This would prevent the possibility of the question 1+H-D from having D greater than 1+H. Line 680 can also indicate if a person got the question wrong by inserting a line 685:

```
685 IF GD>1+H-D*C THEN PRINT "Wrong"
```

I have also enclosed a program which you may find interesting.

Vincent Maker

Ancient History: the battle of Thermopylae, in Extended BASIC

```
100 REM A history quiz by V Maker
```

```
110 REM Dedicated to Mr Johnson
```

```
120 CALL CLEAR
```

```
130 PRINT "History Quiz by V Maker"
```

```
140 FOR T=0 TO 8
```

```
150 PRINT
```

```
160 NEXT T
```

```
170 FOR T=0 TO 200
```

```
180 NEXT T
```

```
190 CALL CLEAR
```

```
200 PRINT "#1 Who led the Spartan forces at  
Thermopylae?"
```

```
210 PRINT "1) Richard Eagan          2) Xerxes  
      3) Pavsomivs                    4)
```

Leonidas"

```
220 CALL KEY(1,K,S)
```

```
230 IF S=0 THEN 220
```

```
240 IF K=9 THEN RIGHT=1 ELSE WRONG=1
```

```
250 IF K=9 THEN PRINT "Right" ELSE PRINT "Wrong"
```

```
260 PRINT "Q2 Who was the traitor at Thermopylae?"
```

```
270 PRINT "1) Ephialties          2) Xerxes  
      3) Themistocles              4)
```

Leonidas"

```
280 CALL KEY(1,K,S)
```

```
290 IF S=0 THEN 280
```

```
300 IF K=19 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
```

```
310 IF K=19 THEN PRINT "Right" ELSE PRINT "Wrong"
```

```
320 PRINT "Q3 Who records the battle of Thermopylae?"
```

```
330 PRINT "1) Herodotus          2) Bradley  
      McPherson          3) Plutarch          4)
```

Thucydides"

```
340 CALL KEY(1,K,S)
```

```
350 IF S=0 THEN 340
```

```
360 IF K=19 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
```

```
370 IF K=19 THEN PRINT "Right" ELSE PRINT "Wrong"
```

```
380 PRINT "Q4 How many Spartans were there at  
Thermopylae?"
```

```
390 PRINT "1) 100          2) 200  
      3) 300                    4)
```

325"

```
400 CALL KEY(1,K,S)
```

```
410 IF S=0 THEN 400
```

```
420 IF K=8 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
```

```
430 IF K=8 THEN PRINT "Right" ELSE PRINT "Wrong"
```

```
440 IF RIGHT=4 THEN A$="Perfect! 4 out of 4"
```

```
450 IF RIGHT=3 THEN A$="Good 3 out of 4"
```

```
460 IF RIGHT=2 THEN A$="Fair 2 out of 4"
```

```
470 IF RIGHT=1 THEN A$="Poor, 1 out of 4"
```

```
480 IF RIGHT=0 THEN A$="Terrible, 0 out of 4"
```

```
490 PRINT A$
```

```
500 IF RIGHT<4 THEN INPUT "Like to try again?(Y/N)":A$  
      ELSE 530
```

```
510 IF A$<>"N" THEN 100
```

```
520 PRINT "Bye!"
```

```
530 END
```

Thanks, Vincent for your ideas and your program.

I hope some of you others are trying out some ideas. I noticed that when I tried out Vincent's latest program that there is a little bug. See if you can find it and suggest a cure for it. Hint: you have to go around the quiz more than once with less than 5 correct answers in total.

As to the improvements to the quiz program, try this idea. When you want to test division, use random numbers to select the answer and the divisor, making sure that the divisor is not zero. Then the dividend can be calculated by multiplying the answer and the divisor and the answer will always be an integer. The same approach can be used with subtraction to make sure the answer is always positive for example. Try some of these ideas out for yourselves. You can also try writing small programs and put them in loops to see how long they take to run 100 times. Then change the program somehow, put more statements on one line for example, and try it again. This is the way to find out more about writing programs if you are interested in that.

Dear Jenny,

I have had another letter from Christopher of Winston Hills.

Dear Crocodile Jones,

I am having trouble with adventure number 2 (Pirate Adventure). How do I get the lumber out of the cavern?

Yours faithfully, Christopher
of Winston Hills.

Well Christopher, what you do is you get the water wings from the shed in the cavern. Leave it on the beach until you have got the bottle from the musty attic after you give it to the pirate in the grass shack and he takes it away. After that you collect the keys from from the rug by pulling the nails away with the hammer and pulling the rug and getting the keys, water wings from the beach then you go to the lagoon. Go north into the sea and get the water and the fish then go to the cave ridden hill. Go into the cave and go down then unlocking the door. Then say "go hall" then you will be able to get the lumber and the shovel. Then if you have your book with you say yoho, if not go out by the maze of caves.

Thanks for the letter. I wish a Merry Christmas to all.

Yours faithfully, Crocodile Jones.

continued from page 30

Archiver III, Lucie Dorais' tutorial on Extended BASIC with a calendar program and Stephen Shaw on BASIC sorting and an introduction to Extended BASIC and its enhancements over BASIC.

The Northern NJ July-August newsletter has an article on improving the video by changing a load resistor on the video processor chip, the TMS9918A. I am not sure whether this will do anything for the PAL chip, the TMS9929A. There is also a description on adding pause, quit and interrupt switches to the console. Treat these with caution also. There are articles on programs for drawings, a rewrite of the real time clock produced by John Willforth from data from John Paine and Gary Emmich (who originated the idea) as well as the extension to add an analog to digital converter and the continuing series from Mickey Schmitt on cassette files.

Geneve (Myarc 9640) - an overview

by Garry J Christensen, TIBUG

One of the new Geneve computers was at last available for demonstration at the last meeting. It was an NTSC version so until we connected it to a compatible monitor the picture was in black and white. The NTSC monitor revealed the colours of the now well known blue swan.

The computer arrived as four components: the card, the keyboard, the manual and the software. All are essential for the operation of the computer. The card is the same size as any other card that fits the expansion box except that it has a part of the board extending out the back to allow access to the ports.

The keyboard is the standard IBM style and plugs into the one of the above-mentioned ports. Two others provide the connections for the joysticks and the mouse. The fourth is the plug for the monitor. It is an 8 pin DIN plug. The pin-outs are given in the manual for the plug.

On disk are the MDOS operating system, MY-WORD, Multiplan upgrade (you need to have the module to make use of this program), the Cartridge Saver that saves modules to disk so that they can be used with the Geneve, a copy of Myarc Extended BASIC II and the GPL interpreter that allows the Geneve to act as a TI99/4A.

Finally the manual covers the setting-up of the computer, the functions of the keys, operation of MDOS, a description of the commands available in the Myarc Advanced BASIC (not completed yet), and a very brief technical reference guide. Several pages of addendum cover the operation of the other programs supplied with the computer.

Installation is simple. Firstly remove the interface card, 32K memory expansion and RAMdisks (if any) from the expansion box, then install the computer. Plug the keyboard and monitor into the back and you are ready to go. The RAMdisks can be used with Geneve but must occupy specific CRU addresses and must be given specific names. One problem that we encountered was that the video output is not compatible with the TI99/4A modulator. If you intend to use a television as a monitor, a special modulator will be required. These will be organised before the PAL versions arrive in the country. If you have a composite video or RGB monitor, there will be no problem.

There is too much to do a full review of every facet of the Geneve in one review, so to start I will be covering the general function of the computer. In the following newsletters I (and others I hope) will be able to give detailed descriptions of individual programs.

When you power-up, the Myarc swan is displayed while the DOS is loading. Even though the DOS occupies a full side of a diskette, it loads surprisingly fast. I believe even faster versions have been produced in the USA.

The majority of the DOS routines are available. Another version has just been released and the new one should be just about complete. From the DOS screen you have three choices: stay and perform DOS operations, enter MY-WORD, or load the GPL interpreter. I will cover the operation of MDOS next month. Type MW and the MY-WORD program loads. Once again, more about MY-WORD in future issues, stay tuned.

The most interesting, at this stage was the GPL interpreter. Type GPL and the program loads and executes. The prompt screen allows you to load a module saved with the CSAVE program. Once you have loaded a module, press enter and you have before you the TI99/4A colour bar screen. From here on it is just like a TI99/4A. Press any key and you have the choice of BASIC or the module that you loaded. It is possible to return to the GPL screen to load another module.

I tried the Editor/Assembler, Extended BASIC and Parsec modules. All functioned normally with the exception of the Extended BASIC. This one will not recognise the FCTN[4] key to break a program. A simple fix is available to cure this problem.

I also loaded some assembly games, from Extended BASIC. Some of these functioned normally, some did not

recognise the keyboard and others recognised the keyboard and not the joysticks. This problem is to be expected in some assembly programs that directly access the keyboard. In the Geneve the keyboard is in a different place. This problem is caused by the use of an IBM keyboard.

I had little time to try out the Myarc Extended BASIC II so I cannot comment on this part of the system. To load the Extended BASIC II you first load the Editor/Assembler module and run a program file (option 5). This loads and executes the Extended BASIC II environment.

The GPL screen gives you the option of selecting the speed, from 1 to 5. 1 is the speed of the TI99/4A, 5 is the fastest. Some programs will not tolerate being run at the faster speeds while others go like greased lightning. It has to be seen to be believed.

I found the Geneve very easy to get used to, particularly the keyboard. The speed of disk access was also increased. The path to run a TI99/4A program tended to be a bit tortuous. Loading the DOS, GPL interpreter, then the module and finally the program takes an appreciable time but as further software is developed, new methods will become available. I would hope that Advanced BASIC will load directly from DOS. That would certainly be the most sensible method of instituting it.

So there you have it. I hope that I have been able to give you a brief understanding of this new machine. I feel that it has a lot of potential to be a great computer. All it needs is the software. If the quality of software that is available for the TI99/A is a guide, the larger, faster computer will be among the best I have seen.

Many questions remain unanswered but the most significant one is: "when will they arrive?" Answer: "Middle of April."

The boards are in production now. Myarc were waiting for me to test the PAL prototype board (partially functioning) and I am pleased to say that it did work. As soon as the production run is complete they will be posted. Allow 3 weeks and I will have them. Then the whole country will know about it.

That is all for this month. See you next month with more news.....Garry

Stop Press.....Myarc have just posted the Hard Disk Controller cards. These are the completed production models, not test versions. The card controls up to 4 floppy drives and up to 3 30Meg hard drives. One limitation of the hard drive system is that the controllers only work with MFM format drives. In Australia, this format is used for most drives up to 20M. The 30M drives use a different system called RLL. These can be used but must be formatted as a 20M drive. Large capacity hard drives can be used but they must be compatible with the MFM format. I have seen drives of 80M advertised but they are big dollars. Delivery date is expected to be early April.

continued from page 28

Disk Manager error codes From the Disk Memory System manual.

#	first-	second-digit
1	other	record not found
2	seek/step	cyclic redundancy code
3	input	lost data
4	print	write protected
5	----	write fault
6	----	no disk or no drive
		or drive not ready
7	----	invalid input parameters
8	----	----
9	special error code for comprehensive test	

continued from page 2

LIBRARY DISK A269 - Some adventures for the TI module or equivalent disk version. Titles are First Days in Eden, Doors to Eden, Discovery at June Lake, On the Loose and Lost Gold. Enough to keep all keen adventurers happy for a long time.

Listing BASIC Programs

by Stephen Shaw, UK

There are many ways of producing program listings, and many ways of manipulating programs. This little article is to show you one program, which has been manipulated with three others, so you will see the original and three variations. Not shown here is the 28 column format, produced by Tony McGovern's COLIST program, as that has been used many times in this magazine already.

COLIST, UNBASHER, and NEATLIST are all available from the User Group disk library, while SMASH is a commercial program available from Tenex.

First the original program. I am only showing a central portion of the program, rather than take up too much magazine space!

```
1440 FOR X1=1 TO 4
1460 R(X1)=ASC(SEG$(P$,X1,1))
1480 NEXT X1
1500 W=0
1520 FOR I=1 TO 4
1540 FOR J=1 TO 4
1560 IF G(I)<>R(J)THEN 1640
1580 W=W+1
1600 R(J)=0
1620 GOTO 1660
1640 NEXT J
1660 NEXT I
1680 W=W-B
1700 RETURN
```

Once this program has been SMASHed, these lines look like this:

```
1440 FOR X1=1 TO 4 :: R(X1)=ASC(SEG$(P$,X1,1)):: NEXT
X1 :: W=0 :: FOR I=1 TO 4 :: FOR J=1 TO 4 :: IF
G(I)<>R(J)THEN 1640.
1580 W=W+1 :: R(J)=0 :: GOTO 1660
1640 NEXT J
1660 NEXT I :: W=W-B :: RETURN
```

Which can be made a little more meaningful by running it through NEATLIST:

```
01440 FOR X1 = 1 TO 4 ::
R(X1) = ASC(SEG$(P$ , X1 , 1)) ::
NEXT X1 ::
W = 0 ::
FOR I = 1 TO 4 ::
FOR J = 1 TO 4 ::
IF G(I) < > R(J) THEN 1640
01580 W = W + 1 ::
R(J) = 0 ::
GOTO 1660
01640 NEXT J
01660 NEXT I ::
W = W - B ::
RETURN
```

VARIABLES USED IN MAIN PROGRAM

(in lines 1440 to 1700 only)

G	01440
I	01440 01660
J	01440 01580 01640
P\$	01440
R	01440
W	01440
X1	01440

(The variable listing is available as an option with the listing, or on its own).

We can also resplit the SMASHed program by using UNBASHER, which produces the following. Note that the output has not been resequenced, but of course it could have been:

```
1440 FOR X1=1 TO 4
1441 R(X1)=ASC(SEG$(P$,X1,1))
1442 NEXT X1
1443 W=0
```

```
1444 FOR I=1 TO 4
1445 FOR J=1 TO 4
1446 IF G(I)<>R(J)THEN 1640
1580 W=W+1
1581 R(J)=0
1582 GOTO 1660
1640 NEXT J
1660 NEXT I
1661 W=W-B
1662 RETURN
```

You may not have a lot of control over the format of a program you purchase or receive from elsewhere, such as the library, but the utilities are available to change the format to something which may be easier to read/debug, or which perhaps occupies a little less memory space and runs a trifle faster..

continued from page 21

```
1040 B=B-(S*NB)
1050 S$=STR$(S)
1060 NB=NB/2
1070 CALL HCHAR(R,C,48+S)
1080 REM
1090 REM *UPDATE SCR LOC*
1100 REM
1110 IF C<32 THEN 1230
1120 C=2
1130 IF R<24 THEN 1220
1140 CALL SOUND(150,1400,0)
1150 CALL KEY(0,K,S)
1160 IF S=0 THEN 1150
1170 REM
1180 REM *CLEAR SCREEN*
1190 REM
1200 CALL VCHAR(1,3,32,720)
1210 R=0
1220 R=R+1
1230 C=C+1
1240 REM
1250 NEXT L
1260 REM
1270 NEXT ADR
1280 CALL SOUND(150,1400,0)
1290 CALL KEY(0,K,S)
1300 IF S=0 THEN 1290
1310 CALL CLEAR
1320 RETURN
```

continued from page 7

Turns all sound generators off in case of false operation due to an output which turns on a sound output.

k) End.

Allows return to Extended BASIC on operation of FCTN[4], (clear).

3. Tutorial day details.

The little train, set on a board 1.1 by 0.9 metres, will be demonstrated at the TIsHUG Tutorial day at Woodstock Community Centre, Church St, Burwood. The doors open at 9am and the tutorial is timed to start at 9.30am after set up is complete. A tutorial booklet will be available from this part of the tutorial day giving circuits, design ideas and assembly program source listings.

There will also be a demonstration of a Forti Music system, (a forthcoming kit from TIsHUG), and Craig Sheehan will present a session on his Extended Display (XDP) package. Russell Welham will have a full system with printer available for members without printers who want to printout a data file or program that they are working on. Members wishing to avail themselves of this service, who use cassette, should bring their own tape recorder as well as the cassette tape to ensure satisfactory loading and any special modules such as PRK, etc.

A light lunch will be provided for a small fee and the TIsHUG shop will be operational.

Remember the date, Saturday 5th November 1988.

Reading the Disk Bit-map

Author unknown

Before anyone can really start "digging around" in TI-DOS, one must first get a handle on the infamous DISK BIT MAP found on sector zero of each TI-DOS disk. In this article we will learn how to read that map, see which sectors are open, and we will do all this without the aid of any of those fancy disk reading programs, we will do it from BASIC!

It is in there!

Everything you need to know about the TI99/4A Disk bit map can be found in VDP memory. Each time you open a file in TI-BASIC, the TI-DOS built into the console (in the disk controller DSR actually, ED) dumps the entire sector zero into VDP RAM. DOS also copies the File Information Block and the latest sector accessed by the BASIC Operating System to boot! All you need to do is open a file and it is all there to see.

Reading sector zero from BASIC

Once a file has been opened, the disk's sector zero is copied into VDP RAM starting at >3EF5 (16117). You will remember from our article this past fall, that the first ten bytes holds the disk name. Other pertinent information like total sectors on disk, the number of sectors per track, number of sides and density are also stored in the first several bytes of sector zero. The rest is used by the DOS to keep track of which sectors are free and which ones are presently in use. When you delete a file, the only action that takes place is the updating of the Bit Map on sector zero. Nothing is erased! This will come in handy when we learn how to recover blown files by changing the map.

What sectors are free?

The Disk Map is written out so that each bit represents a sector. If the sector is in use the bit is set to one (1). If it is free then the bit is set to zero (0). When you first initialize a disk, the bit map is set to >03. (00000011 in binary). This means that sectors zero (the volume Information Block) and sector one (the alphabetical look-up block) are in use.

As each sector is claimed by a data file or a program file, the map is updated accordingly. When a file is "deleted", the appropriate sectors are simply set to zero to indicate that they may now be used by any future file(s). In fact, until some file is actually written over these "free" sectors, all data is still intact. It is very easy to fully recover "deleted" files if no other data has been written on the disk!

The program

The program below reads the Volume Information Block (sector zero) and also displays the disk Bit Map. Since it is written in TI-BASIC, it runs a bit slowly. If you have the CorComp Card or if you have the Miller's Graphics VPEEK, VPoke utilities, translate the program so that it can run in Extended BASIC - it will go lot quicker!

For those of you who like to save handy routines, lines 1010 to 1060 (the end of the loop is really at 1250) contain a very useful method of stripping off bits from a decimal byte value.

NOTE: In order to get the TI Operating System to dump the VIB into VDP RAM, a new file must be created. This program creates a file with the name CHR\$(1) which is not a displayable name. Do not stop the program in mid-stream. At the end of the routine, the file is automatically deleted.

```
100 REM *****
110 REM *
120 REM * VOLUME BLOCK *
130 REM *
140 REM * READER *
150 REM *
160 REM *****
170 REM
180 TB=18
190 MAP1=16173
```

```
200 MAP2=16372
210 REM
220 REM *TITLE SCREEN*
230 REM
240 CALL CLEAR
250 PRINT "DISK VOLUME READER"
260 PRINT "=====
270 PRINT
280 PRINT
290 PRINT
300 INPUT "ENTER DISK# (1-4):":D
310 IF (D>4)+(D<1)THEN 300
320 CALL CLEAR
330 PRINT "DRIVE #";STR$(D);" VOLUME INFO BLOCK"
340 PRINT
350 PRINT "-----
360 PRINT
370 REM
380 OPEN #1:"DSK"&STR$(D)&". "&CHR$(1)
390 REM
400 REM *GET VOLUME INFO*
410 REM
420 VN$=""
430 CALL PEEKV(16117,N(0),N(1),N(2),N(3),N(4),
N(5),N(6),N(7),N(8),N(9))
440 FOR L=0 TO 9
450 VN$=VN$&CHR$(N(L))
460 NEXT L
470 DISPLAY "VOLUME NAME";TAB(TB+1);VN$
480 REM
490 REM *GET DISK INFO*
500 REM
510 CALL PEEKV(16127,TS1,TS2,SPT,I1,I2,I3,P,TPS, NS,DD)
520 REM
530 DISPLAY "TOTAL SECTORS";TAB(TB);256*TS1+TS2
540 DISPLAY "SECTORS/TRACK";TAB(TB);SPT
550 DISPLAY "INIT$";TAB(TB+1);CHR$(I1)&CHR$(I2)&
CHR$(I3)
560 DISPLAY "PROTECTION";TAB(TB+1);CHR$(P)
570 DISPLAY "TRACKS/SIDE";TAB(TB);TPS
580 DISPLAY "# OF SIDES";TAB(TB);NS
590 DISPLAY "DENSITY";TAB(TB);DD
600 PRINT
610 PRINT "-----
620 PRINT
630 INPUT "DISPLAY BIT MAP? (Y/N):":Y$
640 IF Y$<>"Y" THEN 660
650 GOSUB 790
660 CLOSE #1:DELETE
670 REM
680 PRINT
690 INPUT "DO ANOTHER DISK? (Y/N):":Y$
700 IF Y$="Y" THEN 290
710 CALL CLEAR
720 END
730 REM
740 REM *DISPLAY BIT MAP*
750 REM
760 REM
770 REM *SET UP SCREEN*
780 REM
790 CALL CLEAR
800 FOR X=0 TO 23
810 X$=STR$(3*X)
820 IF LEN(X$)>1 THEN 840
830 X$=" "&X$
840 FOR L=1 TO LEN(X$)
850 CALL HCHAR(X+1,L,ASC(SEG$(X$,L,1)))
860 NEXT L
870 NEXT X
880 REM
890 REM *MAIN LOOP*
900 REM
910 R=1
920 C=3
930 FOR ADR=MAP1 TO MAP2
940 REM
950 REM *GET A BYTE*
960 REM
970 CALL PEEKV(ADR,B)
980 REM
990 REM *CALC BITS*
1000 REM
1010 NB=128
1020 FOR L=1 TO 8
1030 S=INT(B/NB)
```

continued on page 20

Extended Business Graphs

by John Ryan

The Extended Business Graphs known as EBG-Files leaves much to be desired. I went to the Pie Chart option and entered 10 numbers ranging from 1 to 10 so that I could find the true % at a glance. The chart looks like this.

Entry	Result	Actual	2 Places.
A 1	1%	1.8181%	1.82
B 2	3	3.6363	3.64
C 3	5	5.4545	5.45
D 4	7	7.2727	7.27
E 5	9	9.0909	9.09
F 6	10	10.9090	10.91
G 7	12	12.7272	12.73
H 8	14	14.5454	14.55
I 9	16	16.3636	16.36
J 10	18	18.1818	18.18
Total	55. 95%	99.9995%	100%

Average is $55/10 = 5.5$ (Graph shows 5.)

Small entries magnify the error, as you will discover later.

% is calculated by (F)

6 divided by total 55 * 100 = 10.909%

It will be appreciated that the programmer has excelled himself with the concept of the graph presentation and ignored accuracy. It appears that all figuring was done with a print integer function (INT). Instead of $INT(NO+.5)/100$ to produce the nearest to 2 decimal places.

As INT disregards numerals to the right of the decimal point, we can manipulate the "point" to suit our requirements. Example --:

If $P = 3.141592654$

INT(P) returns 3

INT(P) returns 31

#INT(P+.5) returns 31

INT(P+.5)/10 returns 3.1

INT(P0+.5)/100 returns 3.14

INT(P00+.5)/1000 returns 3.142

3.1415 is the turning point

3141.5 result of 00

+5

3142.0 is after +5

3.142 results from 3142/1000.

Denotes first time .5 activates.

3.14159=31.4159 returns 31.

+5

31.9159 returns 31.

Having ignored the .9159

Should we $P00 = 3141.59$

+5

3142.09

Here we get a spill over into the whole number. INT ignores the .09 leaving 3142 / by 1000 produces 3.142. The net result is that we have turned Pi into a three decimal number.

With the TI99/4A, one should always let it do all calculations using its power and finally PRINT the result using $INT(P00+.5)/1000$.

Should you run EBG-Files* using one incremented by one up to ten it will be seen that the numbers to the right of the decimal point have been written off. Please note that the Average has been clipped from 5.5 to 5 neat.

Now run EBG-Files** using 1000 to 10,000 and it will give inaccurate percentages as did Files* but the average is now correct, which indicates, that decimal portions of the whole have been discarded during the calculations. This is a pity as the TI99/4A loves to crunch those big numbers.

The Bar Graph sectors on the vertical line have been cut into 1/8 or 12.5% of the whole, 55 being 100%.

We have J at 18.1818% of 55 which is 9.999 and registered as 10 on the graph. Likewise A at 1.8181% of 55 = being just below the line.

E uses 9.09% of 55 (4.995) shown as 5 on the line. X,Y Graph splits the line into 20% of the total i.e. 10,000 top figure is shown as 2, 4, 6, 8, 10 multiplied by 1000. or 20% of 10,000 = 2000 etc.

I would hope that this program would be used only to record rain or other uses not requiring a "spot on" result, as such we will forgive the Author.

PS. The sum of numbers from one to n. is found by--:

```
100 INPUT N
110 N=(N 2+N)*.5
120 PRINT N
130 END
```

continued from page 13

In the Mail Bag section, Dr G Gray writes....

In Hitchhiker's Guide, stop bulldozer will also do, instead of lay down in front of bulldozer.

Games: Well, the book and the series both laid down, so that is why we wrote that. Also the bulldozer can be started up again and then knock your house down, but if you lay in front of it, they cannot touch your house.

Dr Gray: If you wish to get out the air lock, just press the green button.

Games: That also works, but on the series, it happened automatically. But your idea is good, because it saves time.

Dr Gray also wrote much more, but I do not have enough time to share it all with you. Thank you very much Dr Gray for your letter (sorry I did not answer it sooner).

If you are sick of waiting 3 minutes for the Infocom adventures to load, here are two short cuts.

(1) Go to BASIC, with Editor Assembler plugged in and type OLD DSK1,LOAD1

RUN

This is a lot faster.

(2) If you do not have the Editor Assembler Module, you can get a program from the club library (or us), that lets you load the adventure from Extended BASIC in 30 seconds.

Remember if you have any ideas that could help the TI99/4A community, let us know and you can get your name in print, in our column. Please also send adventure hints, clues or anything you like to:

Robert Brown

C/- Games Information

141 Beecroft Road,

Beecroft NSW 2119.

continued from page 6

```
410 IF (K<49)+(K>48+Y) THEN 400 ELSE FN=K+48
420 REM SORTING : FN=FIELD NUMBER
430 REM : FS=FIELDS START : FL=FIELDS LENGTH
440 DISPLAY AT(12,5)ERASE ALL:" Sorting..." :: FS=F(FN)
:: FL=F(FN+1)-FS :: K=X :: L=INT(X/2)+1
450 IF L<>1 THEN L=L-1 :: S$=A$(L) :: GOTO 470
460 S$=A$(K) :: A$(K)=A$(1) :: K=K-1 :: IF K<1 THEN
A$(H)=S$ :: GOTO 350
470 J=L
480 H=J :: J=J+J :: IF J>K THEN A$(H)=S$ :: GOTO 450
490 IF J>=K THEN 510
500 IF SEG$(A$(J),FS,FL)<SEG$(A$(J+1),FS,FL) THEN J=J+1
510 IF SEG$(S$,FS,FL)>SEG$(A$(J),FS,FL) THEN A$(H)=S$
:: GOTO 450
520 A$(H)=A$(J) :: GOTO 480
530 REM SAVE ON DISK INC TAB
540 DISPLAY AT(12,1)ERASE ALL:"Device and filename":
DSK":D$
550 ACCEPT AT(13,6)SIZE(-12):D$ :: IF D$="" THEN 240
ELSE OPEN #1:"DSK"&D$,DISPLAY,VARIABLE 80,OUTPUT
560 FOR Y=0 TO X+2 :: PRINT #1:A$(Y) :: NEXT Y :: CLOSE
#1 :: GOTO 350
570 REM QUIT
580 DISPLAY AT(12,1)ERASE ALL:"<ARE YOU SURE? Y/N>"
590 CALL KEY(3,K,S) :: IF S<>1 THEN 590
600 IF K=89 THEN STOP ELSE IF K=78 THEN 350 ELSE 590
610 DISPLAY AT(22,1):"File not properly organised":
<press enter> :: INPUT D$ :: GOTO 230
620 END
```

Rambles from Great Britain

by Stephen Shaw, UK

Your letters are always welcome, on any subject. Ask me anything you like, I can always say I know nothing, but you will not know if you do not ask! Anything you would like to see in Rambles perhaps? The address follows, and is the same address as the disk based library. Send two disks and return postage for a copy of our disk library list in DV80 format.

10 Alstone Road, STOCKPORT, Cheshire, UK, SK4 5AH

CHANGES, CHANGES.

If you look back to Issue 19, you will see a notice inside the front cover that we are not affiliated to TI etc., etc. That notice was removed from Issue 20. This does NOT mean we are NOW affiliated to TI etc., etc.

This is a history lesson. When the first ever UK TI99/4 User Group was founded, Texas Instruments Inc were extremely protective not only regarding information on their new computer, but also on the use of their name, including the use of the initials TI within the magazine name, which was then TIdings. Their legal department insisted on the disclaimer and who are we to argue with a corporation that big! We are not suggesting that any user group was ever affiliated to TI, we were merely doing what TI insisted we do.

Since those balmy days, TI has every appearance of having lost interest in their home computer, its programs and even their initials! And as one UK user took umbrage at our using the enforced disclaimer, we have dropped it to avoid causing offence. (Yes, I know every good UK citizen regards the initials TI to stand for Tube Investments, but Texas Instruments were very very protective back then!!) You can spot the older User Groups by looking for a non-affiliation disclaimer in their newsletters.

MACHINE CODE FROM TAPE:

Did you know that CALL INIT:: CALL LOAD("CS1") works! Of course first you have to transfer those DF80 files from disk to cassette. And some of the lines may need shortening (you do remember that cassette data files do not use 80 byte records!), dropping off the checksum is a good start there!

```
eg from ..9BD3A87F131F    0001
      ..5A47F196F        0002
      ..A127F1BFF        0003
```

```
shorten to:
      ..9BD3A8F          0001
      ..5A4F             0002
      ..A12F             0003
```

ie remove the 7 tag and the next four characters. Leave the final F. Read each record from disk and transfer the first 64 characters to tape. You will need to note how many lines are to be transferred or use the EOF function. The tape file is saved as DISPLAY, FIXED 64. You will also need to know the starting name to LINK to.

Then of course, there is the time element. Take a look at how many lines your average DF80 file has. A 54 line file, loaded from tape, takes 6 minutes. Now work out how long it is going to take to transfer SUPER DEMON ATTACK this way, and where are you going to find a cassette that long!

Just one more capability the TI99/4A has, that has not been documented.

Now we have the FUNLWEB LOAD program, tape owners can find it much easier to load machine code into their 32K RAMs using memory image files. Much faster.

Editor Assembler module owners can just use the LOAD PROGRAM FILE option and indicate CS1. It really was clever of TI; the ONLY loader that TI supplied to enable cassette owners to load machine code into expansion RAM and it is sold on a high priced package; module, huge manual, and two disks. Thanks to George Meldrum, TISHUG. May 1988.

INFOCOM adventures are not all fully logical, there is a random element in some of them which means that sometimes you die and sometimes you live and an interesting command to type in to your INFOCOM adventures is \$VE try it!

There is a #RAND command in Lurking Horror, which expects a number before/after it, not sure what it does but I think it may determine the path when you come to a random choice. HITCHHIKER has a total vocabulary of 969 words- have you found them all yet? KILL ADAMS? Some odd commands, purpose unknown include YXXY and ZZMGCK, the latter may just be an end of file dummy. SUSPENDED has a vocabulary of 680 words, but you can complete it with just 35, that is real overkill!

MICROPENDIUM:

If you do not yet subscribe to MICROpendium, why not? The cost is reasonable. Prices advertised in March 88 copy (I get mine by seamail!) are US\$23.50 per year by sea mail and US\$37 by airmail. Send US\$ International Money Order, bought from Lloyds or Barclays Banks, to:

MICROPendium, PO Box 1343, Round Rock, TX, USA, 78680.

This is no fly by night magazine. The March 88 issue is their 50th monthly issue. Regular articles on BASIC, c99, Geneve, and plenty of reviews. Back issues are \$2 each by seamail to subscribers only.

MINI MEMORY PROBLEMS?

As the battery wears out, the voltage drops, and in the end, the dead battery actually stops the module working; it will not retain data even inserted in the console, even though the battery is only for "back up". A quick solution is to open the module up and snip one of the battery wires. The module will now work in the console! For battery back up, take advantage of the offer advertised in this magazine for several years now for ni-cad replacements.

ATARISOFT BUG:

In case you just find one, the early Atarisoft modules for the TI99/4A do not all work properly in some consoles. Atari downloaded the "large character set" from grom 0, but used a direct address instead of the indexed address, and yes, some consoles are 4 bits out on the direct address. It means those characters look kinda Japanese.

MILTON BRADLEY BUG (and others):

Some MB games intended to use speech have a bit of a problem when loaded from disk dumps into the 32k RAM. The delay in testing for speech is just not long enough (see EdAs manual pages 349/350), so sometimes the games will fail to identify that you have a speech synthesizer, and give no speech.

TI FORTH DEBUG

The following corrections have just surfaced from Ottawa, passed on without testing. Change a copy of your Master Disk!

```
Screen 58 line 10 to read:
VDPMD 4 < IF SMTN 80 0 VFILL 300 ' SATR ! ENDIF
Screens 53,54,54 line 1 should end SETVDP2.
Screen 59 line 9 change OOFF to OOF
```

TENEX are beginning to upset me. I wrote in Feb 88 asking for a catalogue. No reply. I wrote again in April for a catalogue. No reply. Last chance: wrote again, May 16th 1988. Result: catalogue POSTED May 17th received May 26th. With no envelope or wrapper and nothing holding the outer edges of the pages together. Delivery address and stamps at the bottom of the back cover. Delivery can best be described as a miracle. Fortunately they wrap products up a lot better. If you write, I suggest you ask as strongly as you can that your catalogue is honored with an envelope wrapped around it!!!

Again some nice goodies, and order sent airmail May 31st, including a couple of copies of SPAD XIII for members, who asked for them at the AGM.

Have sent for a freeform database, an alternative spreadsheet, and one or two other goodies, watch for reviews as they arrive here in Stockport.

Prices of many products continue to fall. TENEX have a good range of new items, both disk and module software and peripherals.

For older modules, TEX COMP in California list the most, while PARCO has good stocks of the older modules and even some of the very latest.

=====

CORRECTION: ISSUE 20 : PAGE 42.

Beamheadings Program: Hmmm. Cannot find any 1/2 or 1/4 keys on my keyboard. These odd characters, inserted in the listing by a daisywheel printer, should be characters 123 and 124 respectively, you noticed them redefined at the start of the listing did you not! Sorry about that, program was printed by us in the form it was received!

SECOND HAND PRICES

What is a good price for second hand gear? What somebody will pay for it. Seriously, I have seen adds quoting prices which seem intended to stop anyone even haggling, so high are the prices asked.

If you are selling TI99/4A gear, consider what you would now pay for it yourself! When five hundred will pay for a complete computer with monitor disk drive and everything else, with high resolution graphics, faster processor and so on than the TI99/4A, why should you expect a TI99/4A owner to lay out nearly a thousand on a fully fitted PEB! Everything you are selling is likely to be at least five years old and probably heavily used! May I suggest a limit of 25% of purchase price for second hand goods, many will be worth very much less than this though. New games and educational modules are readily available new for under three pounds! And if any members are giving up their TI99/4As (shame shame) please do consider donating your equipment to the Group, in support of remaining members.

=====

News of further cable problems, this time the hefty cable to the PE Box, which like the Axiom cable is liable to fractured wires if mistreated or subjected to frequent movement. And if your console moves around a lot.

A possible solution. Boots (a large chemist type shop) have opened a section for the disabled (all branches can order all products if not stocked) and one interesting item is a non-slip mat made by Dycem. It seems to be some sort of very grippy plastic, and has its main use as a bottle grip for opening bottles, but also useful for trays and so on. And very suitable to place between a console and a table top, to very effectively stop console movement. Essential if small keyboard users are around, and essential for stand-alone users. Try it!

The Dycem product is not to be confused with so-called non-slip rubber mats you can get for typewriters, which very often are more slippery than nothing! It is very grippy!

=====

As there seems to be a lot of reinventing of wheels going on in the States, with a few "exciting" new offerings not being quite as good as what has gone before, may I draw your attention to programs in the Disk Library which can:

LOAD a DV80 file into a running program FAST.
 RUN a program LISTed to a DV80 file!
 Change compressed/uncompressed DF80 object files to the alternative format.
 Remove the autostart on some DF80 files.
 Embed a DF80 file into an ordinary Extended BASIC program for cassette loading!
 Move a PROGRAM format machine code file onto tape ready to be loaded with Editor Assembler module or, using Funnelweb and Extended BASIC .

Create a PROGRAM format machine code file from a DF80 file, without having to insert SFIRST, SLAST, and SLOAD labels.

Extract lines from an Extended BASIC program fast.

Set an alarm clock to remind you when to finish computing, and so much more.

Take another look at the disk library listing, and advise your needs!

=====

Q: How do I transfer programs from Cassette to Disk?

a. Before loading the tape, type in:

CALL FILES(1)

NEW then

OLD CS1

b. If it loads but will not run, you need to use Extended BASIC. The utility VDPUTIL from the disk library will enable a TIB program to run in Extended BASIC .

c. Send me the tape and a disk and I will make it run from disk, 32K RAM required! Tools available include Myarc Extended BASIC, which is an excellent conversion tool, and procedures to shorten programs.

=====

The disk library has a disk called ED AS ONLY (machine code programs which stupidly insist on using that module!) which has an odd program called WATOR. I have just located the inspiration! Quote:"Sharks and fish fight out their ecological battle on the toroidal planet Wa-Tor, discretely simulating stochastic Lotka-Volterra cycles". Yes, it does that! You input various variables and let the sharks and fishes play. You can read all about it in THE ARMCHAIR UNIVERSE by A K Dewdney, published by W H Freeman. Now all I need is someone to translate the program's input prompts from German to English. Achtung! Volunteer required!

QuestProbe - with SpiderMan

by Larry Saunders

What is an Adventure?

If you have never played an adventure game before, then you are in for a real treat. Adventuring permits the player to move at will within the game environment, and to examine objects for clues that will help reach the object of the game.

For example, an adventure start thus:

I'M IN A ROOM. VISIBLE OBJECTS ARE A RUBY-ENCUSTED BOX AND A CLOSED DOOR. TELL ME WHAT TO DO.

You may want to start by entering a direction (North, South, Up, Down etc.) to see if you can leave the room. Chances are, though, that you will have to find a way to get through the door etc. So, let us try something simple. You type:

OPEN DOOR

But the computer answers in no uncertain terms

SORRY, IT'S LOCKED. WHAT SHALL I DO?

Since the ruby encrusted box appears to be the only other object in the room, let us take a closer look. You type:

TAKE THE RUBY-ENCUSTED BOX

However, the computer responds with:

SORRY, I JUST DON'T UNDERSTAND

Do not despair, pilgrim. The nature of adventuring is such that the computer will generally not understand adjectives, so we must simplify our command and try again. Type:

TAKE THE BOX

This time the computer says:

O.K.

By saying O.K. the computer is telling you that it has understood your command and the box has now been taken. To check this you can type:

INVENTORY

The computer now responds with:

I AM CARRYING: A RUBY-ENCUSTED BOX

Now let us take a look inside:

OPEN THE BOX

O.K.

EXAMINE THE BOX

O.K. THERE IS A KEY AND A RARE POSTAGE STAMP.

continued on page 26

Tips from the Tigercub #51

by Jim Peterson, Tigercub Software, USA

Copyright 1988
Tigercub Software
156 Collingwood Ave.
Columbus, OH 43213

Distributed by Tigercub Software to TI99/4A Users Groups for promotional purposes and in exchange for their newsletters. May be reprinted by non-profit users groups, with credit to Tigercub Software.

Over 120 original programs in BASIC and Extended BASIC, available on cassette or disk, now reduced to just \$1 each, plus \$1.50 per order for cassette or disk and packing and mailing. Minimum order of \$10. Cassette programs will not be available after my present stock of blanks is exhausted. The Handy Dandy series, and Color Programming Tutor, are no longer available on cassette. Descriptive catalogues, while they last, \$1 which is deductible from your first order.

Tigercub Full Disk Collections, reduced to \$5 post paid. Each of these contains either 5 or 6 of my regular catalog programs, and the remaining disk space has been filled with some of the best public domain programs of the same category. I am not selling public domain programs - they are a free bonus!

TIGERCUB'S BEST, PROGRAMMING TUTOR, PROGRAMMER'S UTILITIES, BRAIN GAMES, BRAIN TEASERS, BRAIN BUSTERS!, MANEUVERING GAMES, ACTION GAMES, REFLEX AND CONCENTRATION, TWO-PLAYER GAMES, KID GAMES, MORE GAMES, WORD GAMES, ELEMENTARY MATH, MIDDLE/HIGH SCHOOL MATH, VOCABULARY AND READING, MUSICAL EDUCATION, KALEIDOSCOPIES AND DISPLAYS

NUTS & BOLTS Disks

These are full disks of 100 or more utility subprograms in MERGE format, which you can merge into your own programs and use, almost like having another hundred CALLS available in Extended BASIC. Each is accompanied by printed documentation giving an example of the use of each. NUTS & BOLTS (No. 1) has 100 subprograms, a tutorial on using them and 5 pages of documentation. NUTS & BOLTS No. 2 has 108 subprograms and 10 pages of documentation. NUTS & BOLTS #3 has 140 subprograms and 11 pages of documentation. Now just \$15 each, postpaid.

Tips from the Tigercub

These are full disks which contain the programs and routines from the Tips from the Tigercub newsletters, in ready-to-run program format, plus text files of tips and instructions. TIPS (Vol. 1) contains 50 original programs and files from Tips newsletters No. 1 through No. 14. TIPS Vol. 2 contains over 60 programs and files from Nos. 15 to 24. TIPS Vol. 3 has another 62 from Nos. 25 through 32. TIPS Vol. 4 has 48 more from issues No. 33 through 41. Now just \$10 each, postpaid.

```
*****
*           NOW READY           *
*   Tips from Tigercub Vol. 5   *
*   Another 49 programs and files *
*   from issues No. 42 through 50. *
*   Also $10 packing and postpaid *
*****
```

Tigercub Care disks #1,#2,#3 and #4.

Full disks of text files (printer required). No. 1 contains the Tips news letters #42 to #45, etc. Nos. 2 and 3 have articles mostly on Extended BASIC programming. No. 4 contains Tips newsletters Nos. 46 to 52. These were prepared for user group newsletter editors but are available to anyone else for \$5 each postpaid.

I believe this word game is totally different from anything you have ever seen, and very challenging if you do not use the AID key. The first time you run it, pick option 3 to create a file of phrases and give it the file name COMPUTE. This will then become the computer's file, option 1, and you can create as many of your own files as you want. I recommend phrases of several to as many as 20 words - short ones are too difficult.

```
100 DIM W$(20):: DIM D$(20)
110 GOTO 150
120 Q$=K,S,Q,F$,E,FLAG,X,J,X$,Y$,A,B, M$,DY$,V,A$(),
    C$,CH,CH$,Y,W$(),L,M,D$(),F,Z,C,R,H
130 CALL CHAR :: CALL KEY :: CALL SOUND :: CALL CLEAR
    :: CALL CHARPAT :: CALL COLOR :: CALL SCREEN ::
    CALL VCHAR :: CALL SPRITE :: CALL LOCATE ::
    CALL DELSPRITE
140 !@P-
150 CALL CHAR(94,"3C4299A1A199423C")::
    DISPLAY AT(2,1)ERASE ALL:"TIGERCUB SHUTTLESEARCH
    V.1.1":"":c Tigercub Software for
    free:"distribution but no price"
160 DISPLAY AT(6,1):"or copying fee to be
    charged":"":If you should feel moved to:"send me
    a few bucks for my":"work, I won't be offended!"
170 DISPLAY AT(12,1):"Jim Peterson":"156 Collingwood
    Ave.":"Columbus, OH 43213"
180 DISPLAY AT(16,5):"Instructions? (Y/N) N" ::
    ACCEPT AT(16,25)SIZE(-1)VALIDATE("YN"): Q$ :: IF
    Q$="N" THEN 260
190 DISPLAY AT(2,1)ERASE ALL:" The computer will
    display a":"phrase or saying concealed":"within a
    grid of random":"letters."
200 DISPLAY AT(6,1):" The words will be horizon-":"tal,
    one word per line and":"on consecutive lines,
    but":"not necessarily beginning on"
210 DISPLAY AT(10,1):"the top line, and the
    phrase":"may 'wrap around' from the":"bottom row to
    the top."
220 DISPLAY AT(13,1):" You can find the phrase
    by":"shuttling columns of letters":"up and down,
    looking for":"consecutive rows with letter"
230 DISPLAY AT(17,1):"combinations that could
    be":"parts of words.":" A cheat key is
    available,":"if you are really stuck, but"
240 DISPLAY AT(21,1):"try not to use it!"
250 DISPLAY AT(23,8):"PRESS ANY KEY" ::
    DISPLAY AT(23,8):"press any key" ::
    CALL KEY(0,K,S):: IF S=0 THEN 250
260 DISPLAY AT(3,2)ERASE ALL : "Do you want to - 1":"":
    (1) Solve a saving from my file?":" (2)
    Solve a phrase from your file?"
270 DISPLAY AT(11,2):"(3) Create a file of":"
    phrases?":" (4) Have someone type in a
    phrase to solve?"
280 ACCEPT AT(3,19)SIZE(-1)VALIDATE(DIGIT):Q :: IF Q<1
    OR Q>4 THEN 280
290 ON Q GOTO 300,310,410,470
300 F$="1.COMPUTE" :: E=1 :: GOTO 320
310 DISPLAY AT(18,1):"Filename? DSK" ::
    ACCEPT AT(18,14):F$ :: E=2
320 ON ERROR 370
330 IF FLAG=1 THEN 350 :: FLAG=1 ::
    OPEN #1:"DSK"&F$,FIXED,RELATIVE,INPUT :: ON ERROR
    STOP
340 INPUT #1,REC 0:X :: CLOSE #1 :: FOR J=1 TO X ::
    X$=X$&CHR$(J):: NEXT J :: Y$=X$
350 RANDOMIZE :: A=INT(RND*LEN(Y$)+1)::
    B=ASC(SEG$(Y$,A,1))::
    Y$=SEG$(Y$,1,A-1)&SEG$(Y$,A+1,255):: IF LEN(Y$)=0
    THEN Y$=X$
360 OPEN #1:"DSK"&F$,FIXED,RELATIVE,INPUT :: ON ERROR
    STOP :: INPUT #1,REC B:M$ :: CLOSE #1 :: GOTO 490
370 FOR J=1 TO 10 :: DISPLAY AT(20,1):"" ::
    DISPLAY AT(20,1):"CANNOT OPEN FILE!" ::
    CALL SOUND(-99,110,5,-4,5):: NEXT J
380 ON ERROR 390 :: CLOSE #1
390 FLAG=0 :: INPUT "CHECK DISK AND DRIVE, PRESS ANY
    KEY":DY$
400 IF E=1 THEN RETURN 260 ELSE IF E=2 THEN RETURN 310
    ELSE RETURN 410
410 DISPLAY AT(8,1)ERASE ALL:"Filename? DSK" ::
    ACCEPT AT(8,14):F$
420 E=3 :: ON ERROR 370 :: OPEN #1:"DSK"&F$,FIXED
    124,RELATIVE,OUTPUT :: ON ERROR STOP :: X=0
430 DISPLAY AT(12,1):"Enter END when
    finished":"":"Type phrases, not more than 20
    words and 124 characters"
440 X=X+1 :: ACCEPT M$ :: IF LEN(M$)>124 THEN PRINT
    "TOO LONG!" :: X=X-1 :: GOTO 440
450 IF M$<>"END" THEN PRINT #1,REC X:M$ :: GOTO 440
460 PRINT #1,REC 0:X :: CLOSE #1 :: GOTO 260
470 CALL KEY(3,K,S):: DISPLAY AT(12,1)ERASE ALL:"Type a
    phrase of less than 20 words and press Enter"
480 ACCEPT M$ :: CALL CLEAR
```

```

490 DISPLAY AT(3,2)ERASE ALL:"Choose skill level -
1","" (1) All words begin in":" first column"
500 DISPLAY AT(8,2):"(2) All words begin in same":"
column":" (3) Each word may appear in":"
a different column"
510 DISPLAY AT(14,2):"(4) As No. 3 but AID key
is":" disabled","" (5) Quit"
520 ACCEPT AT(3,23)SIZE(-1)VALIDATE(DIGIT):V :: IF V<1
OR V>5 THEN 520 :: IF V=5 THEN CALL CLEAR :: STOP
530 DISPLAY AT(12,6)ERASE ALL:"SCRAMBLING....."
540 A$(1)="jkzae klmpv vqah nceci sdufy bqijw astrf
urdsa nvjxe blbig trakv nobth wehey vnijo oherq
umbmi rtika opleg nosve tarkh zeski "
550 A$(2)="!boiu m.fgt krac, pjip? tn-un osheg kar,q
ibl.o tons! idrix ?uhig ebarf uks,k ,,jhge vifyt
kibrn taga, !ry lakle ilf.! inst"
560 C$=A$(1)&A$(2)
570 FOR CH=65 TO 90 :: CALL CHARPAT(CH,CH$)::
CALL CHAR(CH+32,CH$):: NEXT CH ::
CALL CHAR(42,"82444428281010")
580 CALL CHAR(143,"18243C4A4A3C2418")::
CALL COLOR(14,16,1)
590 M$=M$&" " :: Y=1
600 X=POS(M$," ",1):: W$(Y)=SEG$(M$,1,X)::
L=LEN(W$(Y)): M=MAX(M,L):: RANDOMIZE ::
W$(Y)=W$(Y)&SEG$(C$,INT(230*MRND+1),20-L)
610 Y=Y+1 :: IF Y=21 THEN 620 :: M$=SEG$(M$,X+1,255)::
IF LEN(M$)>0 THEN 600
620 FOR J=Y TO 20 :: W$(J)=SEG$(C$,INT(230*MRND+1),20)
:: NEXT J
630 ON V GOTO 670,640,650,650
640 X=INT(RND*(20-M))+M+1 :: FOR J=1 TO Y ::
W$(J)=SEG$(W$(J),X,255)&SEG$(W$(J),1,X-1):: NEXT J
:: GOTO 670
650 FOR J=1 TO Y :: X=INT(RND*(20-M))+M+1 ::
W$(J)=SEG$(W$(J),X,255)&SEG$(W$(J),1,X-1):: NEXT J
:: GOTO 670
660 ! the string
670 FOR J=1 TO 20 :: FOR L=1 TO 20 ::
D$(J)=D$(J)&SEG$(W$(L),J,1):: NEXT L :: NEXT J
680 IF V=1 THEN F=M ELSE F=20
690 FOR J=1 TO F :: Z=INT(20*MRND+1)::
D$(J)=SEG$(D$(J),Z,255)&SEG$(D$(J),1,Z-1):: NEXT J
700 CALL CLEAR :: CALL SCREEN(5):: FOR S=1 TO 13 ::
CALL COLOR(S,5,16):: NEXT S ::
CALL VCHAR(1,31,1,96)
710 CALL VCHAR(4,5,143,20):: CALL VCHAR(4,28,143,20)
720 FOR C=1 TO 20 :: FOR R=1 TO 20 ::
CALL VCHAR(R+3,C+6,ASC(SEG$(D$(C),R,1))) :: NEXT R
:: NEXT C
730 DISPLAY AT(1,1):"s&d to select, e&x to scrollFCTN 7
aid, FCTN 8 restart"
740 H=1 :: C=48 :: CALL SPRITE(#1,42,7,18,C)
750 CALL KEY(3,K,S):: IF S=0 THEN 750 ELSE ON
POS("EXSD")&CHR$(1)&CHR$(6),CHR$(K),1)+1 GOTO
750,800,810,820,830,760,840
760 IF V=4 THEN 750
770 FOR S=5 TO 8 :: CALL COLOR(S,16,5):: NEXT S
780 CALL KEY(3,K,S):: IF S=-1 THEN 780
790 FOR S=5 TO 8 :: CALL COLOR(S,5,16):: NEXT S :: GOTO
750
800 D$(H)=SEG$(D$(H),2,19)&SEG$(D$(H),1,1):: FOR R=1 TO
20 :: CALL VCHAR(R+3,H+6,ASC(SEG$(D$(H),R,1)))::
NEXT R :: GOTO 750
810 D$(H)=SEG$(D$(H),20,1)&SEG$(D$(H),1,19):: FOR R=1
TO 20 :: CALL VCHAR(R+3,H+6,ASC(SEG$(D$(H),R,1)))::
NEXT R :: GOTO 750
820 C=C-8-(C=48)*8 :: H=C/8-5 :: CALL LOCATE(#1,18,C)::
GOTO 750
830 C=C+8+(C=200)*8 :: H=C/8-5 ::
CALL LOCATE(#1,18,C):: GOTO 750
840 CALL CLEAR :: FOR J=1 TO 20 :: D$(J)=" " :: NEXT J
:: M=0 :: CALL DELSPRITE(#1):: IF Q=1 OR Q=2 THEN
350 ELSE 470

```

Here are three screen display subprograms of the type you will find on my Nuts and Bolts disks. Note that subprograms can read DATA from the main program. The double colons in the DATA statement cause input of null strings of data for spacing between the lines. The M\$() in the subprogram parameter lists is necessary, even though the array is not passed from the main program, in order to DIMension the array in the subprogram - unless you prefer to place the DIM in the subprogram itself. T is the number of DATA items to be read.

```

100 CALL CLEAR
110 DATA THIS IS A DEMO,,OF THREE SCREEN
PRINTING,,SUBPROGRAMS PUBLISHED IN,,TIPS FROM THE
TIGERCUB,,No. 51,,BY TIGERCUB SOFTWARE
120 DIM M$(11):: CALL DOWNPRINT(M$( ),11):: FOR D=1 TO
1000 :: NEXT D :: CALL CLEAR :: RESTORE 110 ::
CALL DIAGPRINT(M$( ),11)
130 FOR D=1 TO 1000 :: NEXT D :: CALL CLEAR :: RESTORE
110 :: CALL INWARD(M$( ),11)
1000 SUB DOWNPRINT(M$( ),T)
1001 FOR J=1 TO T :: READ M$(J):: L=INT(LEN(M$(J))+.5)
:: M$(J)=RPT$( " ",14-INT(L/2))&M$(J)::
M$(J)=M$(J)&RPT$( " ",28-LEN(M$(J))):: NEXT J
1002 FOR J=1 TO 28 :: FOR L=1 TO T
1003 DISPLAY AT(L,1):SEG$(M$(L),1,J):: NEXT L
1004 NEXT J :: SUBEND
2000 SUB INWARD(M$( ),T):: FOR J=1 TO T :: READ M$(J)::
NEXT J :: R=1 :: FOR A=1 TO T
2001 L=INT(LEN(M$(A))):: F=13-L/2 :: G=L+F
2002 FOR J=1 TO INT(L/2+.5)::
DISPLAY AT(R,F+1):SEG$(M$(A),J,1)::
DISPLAY AT(R,G):SEG$(M$(A),L-J+1,1):: F=F+1 ::
G=G-1 :: NEXT J :: R=R+1 :: NEXT A :: SUBEND
3000 SUB DIAGPRINT(M$( ),T):: FOR J=1 TO T :: READ
M$(J):: L=INT(LEN(M$(J))+.5)::
M$(J)=RPT$( " ",14-(L/2))&M$(J)::
M$(J)=M$(J)&RPT$( " ",28-LEN(M$(J))):: NEXT J
3001 FOR J=1 TO 28+L :: FOR L=1 TO T
3002 IF J<L THEN 3007
3003 DISPLAY AT(L,1):SEG$(M$(L),1,J-L):: NEXT L
3004 NEXT J :: SUBEND

```

Just in case you did not know - to jump directly to the first or last line in a TI-Writer file, use FCTN 9 and S(earch) and l for the first line or E for the last.

Memory almost full... Jim Peterson

continued from page 24

TAKE THE KEY
then:
UNLOCK THE DOOR
Computer responds:
O.K. THE DOOR OPENS
At last you are out and the first obstacle has been successfully overcome. You are on your way.

Some useful new features
There are several new features which have been introduced in Spider-man to help you interact more easily with your computer.

- Stringing together of more than one command using 'THEN' or a comma (,), e.g.
GET THE BOX, OPEN THE BOX THEN TAKE THE KEY
- The use of full sentences, e.g.
GO ALL THE WAY UP.
TAKE THE GEM FROM THE AQUARIUM.
TALK TO MADAME WEB.
- The ability to take or drop more than one object at a time, e.g.
TAKE EVERYTHING
DROP THE GEMS
DROP ALL

Some helpful words
Although, the vocabulary accepted by your computer is extensive, you may find the words listed below to be of some use as you set about your adventure. Remember these are just a few of the many words available.
CLIMB MOVE TALK TAKE PUSH DROP EXAMINE
GO ENTER READ LEAVE QUIT SAVE LOOK
OPEN TURN PULL RAISE DIG JUMP LISTEN
Finally, the computer can understand much more than you think, so experiment.

One letter commands
Z -turn on/off graphics mode(Graphic disk version)
ENTER-review text window(Graphic disk version)
N,S,E,W,U,D-go north,south,east,west,up or down
L -look
I -take inventory of items carried

The Forth Column

Forth Forum <7>

by George L. Smyth

This month I will begin a discussion of the finite loop, a looping structure which has a definite starting and ending point. Also, a couple more comparison operators will be mentioned. To finish things off, a quick way to add speech capabilities to the language will be presented. We are now at the point where we can begin writing simple programs using the words so far presented, and also explore the Forth manual, which seemed so mysterious and esoteric at first.

Finite loops

A finite loop is a looping structure in which the number of repetitions is determined by a count specified by the programmer. An example of finite loop in Basic is the FOR..NEXT construct. FOR X = 1 TO 10 :: NEXT X would loop ten times. The Forth structure which relates to this is the DO..LOOP construct. This is also an example of the use of Forth's other stack, the return stack. (Yes, Forth has two stacks: a parameter stack and a return stack. I will elaborate in a moment).

To use the DO..LOOP construct, the high and low index are placed on the stack before the word DO. The two numbers are moved by the system to the return stack and used as an index for determining whether or not the looping should continue. An example followed by an explanation will probably be most useful. Suppose that we wanted to print the numbers 0 to 9 on the screen. A word to do that might be:

```
: PRT_0-9 10 0 DO I . LOOP ;
```

The first thing that you probably noticed was that I placed the numbers 10 and 0 on the stack. Keep that thought on hold for just a moment. The next thing you noticed was the word 'I'. The word 'I' copies the value from the top of the return stack and places it on the parameter stack. Okay, let us follow what happens.

After the newly defined word is read, the number 10 is placed on the stack (whenever I refer to "the stack", I mean the parameter stack), with the number 0 placed on the stack above it. The word 'DO' takes the top two numbers on the stack and places them on the return stack in the same order. The word 'I' takes the number at the top of the return stack, which at this point is '0', and copies it to the parameter stack. The word '.' prints the number on the top of the stack to the selected output device, the CRT. The word 'LOOP' decides whether or not to leave the loop. It increments the value at the top of the return stack and compares it to the value just below it. If it is equal or larger, it leaves the loop. So, the first time through, the '0' on the top of the return stack would be incremented to '1' and compared to the '10'. Since it is smaller, control would be passed back to the word following 'DO', which is 'I'. The value now at the top of the return stack is '1' and when copied to the parameter stack, would be printed to the screen by '.'. The loop continues until the number '9' is printed. At this point, the top of the return stack, '9', is incremented to '10' and compared with the value below it, '10'. Since they are equal, the loop is left, and in this case, the word has been executed completely.

'I' is one of several words that manipulate the values on the return stack. Two additional words are '>R' and 'R>'.

>R (n ---) Removes (not copies) a number from the top of the parameter stack and places it on the return stack.

R> (--- n) Removes (not copies) a number from the top of the return stack and places it on the parameter stack.

The Forth system itself uses the return stack for a few things while the program is running, so unless you are doing something very tricky, the word '>R' should always be balanced with 'R>' somewhere along the line. These words are used to allow the return stack to be used as temporary(!) storage for values. Now you should be able to go back to last month's "The Forum"

section and follow the word X/MOD. Remember, if you leave values on the return stack without taking them off, you will lock the system up.

Just as the Basic FOR..NEXT construct can be incremented by a value other than "1" by using the word STEP, so Forth can do the same by using the word +LOOP.

+LOOP (n ---) Increments the value on the top of the return stack by the value on the parameter stack during execution of a DO..+LOOP construct.

Let us modify the word PRT_0-9 to print the even values only.

```
: PRT_0-9 10 0 DO I . 2 +LOOP ;
```

In this case, after the "." prints the value placed on the stack by "I", the number "2" is placed on the stack. +LOOP takes this number and increments the value on the top of the return stack, 0, by "2", leaving the value "2". Next time around, this number "2" is copied to the parameter stack and printed to the screen.

A negative number can also be placed on the stack in front of the +LOOP to decrement the value on the return stack.

```
: PRT_10-1 0 10 DO I . -1 +LOOP ;
```

Here we are placing a -1 for the +LOOP word to add to the value at the top of the return stack, which will be "10". The "10" is decremented and "9" is printed to the screen the next time around.

LEAVE (---) The word LEAVE allows you to leave a DO..LOOP construct and continue with the word following LOOP. Often, an IF..THEN test is placed within the structure to allow escaping the loop. A quick example:

```
: READ_SEC/1 200 100 DO I OFFSET DUP 0= IF LEAVE THEN 2 +LOOP INVERT
```

This word is used in my catalog program. The word OFFSET is a word I wrote which reads a word and places its value on the stack. This value is the pointer to a file's location on a disk. If the value is '0', the end of list of files has arrived. I tested this value by copying the value on the stack and comparing it with the number '0' using the word '0=' (I will explain this word in a minute). If I had not reached the end of the list of files, the Boolean value returned by '0=' will be false and the IF..THEN routine will be "skipped". If the value on the stack is '0', a positive Boolean value will be returned, and the word 'LEAVE' will be executed, thus jumping me out of the loop and executing the word 'INVERT' (INVERT is a word I wrote which inverts the stack, that is the value on the top becomes the value on the bottom, etc.). I wrote this word this way because I had no other way of knowing how many files were to be found on each disk. In month or two I will put the entire program in this column.

0= (n --- f) Compares the value on the top of the stack to "0" and returns a true flag if the number is equal to zero, and a false flag if it is not.

OK (n --- f) Returns a true flag if the number on the top of the stack is negative, and a false flag if it is not.

These two words, as indicated, check the value on the top of the stack and return a flag if the comparison they are performing is true. As noted in the word above, these words are very useful when setting up a flag for word whose execution depends upon a flag it expects to find on the stack.

Next month I will go over infinite loops, of which our Forth operating system happens to be an example. I will also try to fill in a couple of things I have left out. At this point the reader should be able to write a couple of simple programs. Needless to say, there is plenty still to cover, but just as we all were able to start writing simple Basic programs in a few days, we are now at this point in Forth. Readers should also begin looking through the user's manual to discover things for themselves. When I first looked at my manual, I was unable to tell which end was up. However, after learning the elementary concepts, I began to understand some of the gobbledygook. Try looking at it now and see if some of it is starting to make some sense.

Error Codes

Author unknown, edited by Geoff Trott

Extended BASIC error codes

These error codes are listed in the Extended BASIC manual, Appendix N. The number is returned by a CALL ERR which may be used to decide what the program should do on the occurrence of a particular error.

```

10 NUMERIC OVERFLOW
14 SYNTAX ERROR
16 ILLEGAL AFTER SUBPROGRAM
17 UNMATCHED QUOTES
19 NAME TOO LONG
20 UNRECOGNIZED CHARACTER
24 STRING-NUMBER MISMATCH
25 OPTION BASE ERROR
28 IMPROPERLY USED NAME
36 IMAGE ERROR
39 MEMORY FULL
40 STACK OVERFLOW
43 NEXT WITHOUT FOR
44 FOR-NEXT NESTING
47 MUST BE IN SUBPROGRAM
48 RECURSIVE SUBPROGRAM CALL
49 MISSING SUBEND
51 RETURN WITHOUT GOSUB
54 STRING TRUNCATED
56 SPEECH STRING TOO LONG
57 BAD SUBSCRIPT
60 LINE NOT FOUND
61 BAD LINE NUMBER
62 LINE TOO LONG
67 CAN'T CONTINUE
69 COMMAND ILLEGAL IN PROGRAM
70 ONLY LEGAL IN A PROGRAM
74 BAD ARGUMENT
78 NO PROGRAM PRESENT
79 BAD VALUE
81 INCORRECT ARGUMENT LIST
83 INPUT ERROR
84 DATA ERROR
97 PROTECTION VIOLATION
109 FILE ERROR
130 I/O ERROR
135 SUBPROGRAM NOT FOUND

```

Editor/Assembler error codes

Extended BASIC error equates

These values are from the Editor/Assembler manual appendices. It is not clear when they would be used, perhaps they are returned as arguments for a BLWP @ERR when using Extended BASIC.

Name	Value	Extended BASIC error and code
ERRNO	>0200	2 Numeric overflow
ERRSYN	>0300	3 Syntax error
ERRIBS	>0400	4 Illegal after subprogram
ERRNQS	>0500	5 Unmatched quotes
ERRNTL	>0600	6 Name too long
ERRSNM	>0700	7 String-number mismatch
ERROBE	>0800	8 OPTION BASE error
ERRMUV	>0900	9 Improperly used name
ERRIM	>0A00	10 Image error
ERRMEM	>0B00	11 Memory full
ERRSO	>0C00	12 Stack overflow
ERRNWF	>0D00	13 NEXT without FOR
ERRFNN	>0E00	14 FOR-NEXT nesting
ERRSNS	>0F00	15 Must be in subprogram
ERRRSC	>1000	16 Recursive Subprogram call
ERRMS	>1100	17 Missing SUBEND
ERRRWG	>1200	18 RETURN without GOSUB
ERRST	>1300	19 String truncated
ERRRBS	>1400	20 Bad subscript
ERRSSL	>1500	21 Speech string too long
ERRLNF	>1600	22 Line not found
ERRBLN	>1700	23 Bad line number
ERRLTL	>1800	24 Line too long
ERRCC	>1900	25 Cannot continue
ERRCIP	>1A00	26 Command illegal in a program
ERROLP	>1B00	27 Only legal in a program
ERRBA	>1C00	28 Bad argument
ERRNPP	>1D00	29 No program present
ERRBV	>1E00	30 Bad value
ERRIAL	>1F00	31 Incorrect argument list

```

ERRINP >2000 32 Input error
ERRDAT >2100 33 Data error
ERRFE >2200 34 File error
ERROR >2400 36 I/O error
ERRSNF >2500 37 Subprogram not found
ERRPV >2700 39 Protection violation
ERRINV >2800 40 Unrecognised character
WRNNO >2900 41 Numeric overflow
WRNST >2A00 42 String truncated
WRNNPP >2B00 43 No program present
WRNINP >2C00 44 Input error
WRNIO >2D00 45 I/O error

```

Execution errors

These also come from the Editor/Assembler manual, but are for TI BASIC and are used by the BLWP @ERR processor to allow reporting of errors by BASIC.

0-7	STANDARD I/O	15	STRING/NUMBER MISMATCH
08	MEMORY FULL	16	BAD ARGUMENT
09	INCORRECT STATEMENT	17	BAD SUBSCRIPT
0A	ILLEGAL TAG	18	NAME CONFLICT
0B	CHECKSUM ERROR	19	CANNOT DO THAT
0C	DUPLICATE DEFINITION	1A	BAD LINE NUMBER
0D	UNRESOLVED REFERENCE	1B	FOR NEXT ERROR
0E	INCORRECT STATEMENT	1C	I/O ERROR
0F	PROGRAM NOT FOUND	1D	FILE ERROR
10	INCORRECT STATEMENT	1E	INPUT ERROR
11	BAD NAME	1F	DATA ERROR
12	CANNOT CONTINUE	20	LINE TOO LONG
13	BAD VALUE	21	MEMORY FULL
14	NUMBER TOO BIG	22-FF	UNKNOWN ERROR

Loader error codes

The TI BASIC loader (or LOAD AND RUN option), generates the following error codes. From the Linking Loader Chapter of the Editor/Assembler manual.

```

0-7 standard I/O
8 memory overflow
10 illegal tag
11 checksum error
12 unresolved reference

```

TI BASIC error codes pertaining to disk system

From the Disk Memory System manual.

#	first-	second-digit
0	OPEN	cannot find specified disk drive
1	CLOSE	device or file is write protected
2	INPUT	bad open attribute
3	PRINT	illegal operation
4	RESTORE	disk full or too many files opened
5	OLD	attempt to read past EOF
6	SAVE	device error
7	DELETE	file error
9	EOF	

TI-Writer error codes

From the TI-Writer manual.

0	INDICATES DISK CONTROLLER NOT ON
	or DISKETTE NOT INITIALIZED
6	NO DISK IN DRIVE DSK1 or DISK UPSIDE DOWN
	or DRIVE IS NOT TURNED ON
7	NO DISK IN DRIVE DSK1
00	ILLEGAL USE OF LoadF, PrintF, SaveF
02	NO FILE ON DISK WITH FILENAME USED
04	DISK IS FULL
06	PrintF COMMAND IN PROGRESS WAS INTERRUPTED
	or DISK DOOR WAS OPENED WHEN LIGHT WAS ON
07	INVALID FILENAME (NAME TOO LONG
	or INVALID CHARACTERS)
15	INVALID DISK DRIVE NUMBER or DEVICE

Input/Output (I/O) error codes

#	first-	second-digit
1	OPEN	DEVICE NOT FOUND
2	CLOSE	WRITE PROTECTED
3	PRINT	INVALID I/O COMMAND
4	RESTORE	OUT OF SPACE
5	OLD	EOF
6	SAVE	DEVICE ERROR
7	DELETE	FILE MISMATCH or DATA MISMATCH

continued on page 19

File Types for the TI99/4A

by Irwin Hott, USA

This article will be concerned with the types of disk files available on the TI. They are: DISPLAY VARIABLE, DISPLAY FIXED, INTERNAL VARIABLE, INTERNAL FIXED and PROGRAM.

We will start looking at PROGRAM files. When you think of a PROGRAM file, you probably think of a BASIC or Extended BASIC one. There are several other types of PROGRAM files. There are a few indicators you can use to help you to decide whether the PROGRAM files on your disk are BASIC, Editor/Assembler, GramKracker etc. BASIC or Extended BASIC PROGRAM files may be anywhere from 2 to 52 or so sectors. After a point which is determined by how much memory is needed, Extended BASIC programs are saved as INTERNAL VARIABLE 254. These programs cannot be run in BASIC. You may be able to save an INTERNAL VARIABLE 254 program from disk to cassette if it is close to the point where it would be saved as a PROGRAM file.

You may also run into Extended BASIC PROGRAM files that are, say 25 sectors long with just a few program lines.

These most likely contain embedded assembly files saved with ALSAVE or SYSTEX. These will not run in BASIC. If you save them in MERGE format, you may lose assembly files.

33 sector PROGRAM files may be Editor/Assembler option 5. If you try to run an Editor/Assembler option 5 in BASIC or Extended BASIC you will get an error 50. Another indication of Editor/Assembler option 5 is that the last character of the file name will be incremented by one ASCII value as in TERM TERN TERO or MGR1 MGR2. In most cases Editor/Assembler option 5 files will be 33 sectors in length except for the last one. There are some exceptions with Editor/Assembler option 5 loaders however.

Files for the GramKracker follow a somewhat similar format except that they are 34 sectors in length. For the GramKracker you may see file names such as: XB XB1 XB2 XB3 XB4 XB5 XB6 or FAST/61 FAST/62 or GRAM0 GRAM1 GRAM2. Some RLE pictures may also be in PROGRAM files. I have seen 25 and 54 sector format pictures. Just a couple of other notes on PROGRAM files. There will soon be a program in the CONNI library which is supposed to make it possible to transfer long PROGRAM files from cassette to disk. Also from Funnelweb Farm, Cassette Transfer makes it possible for you to store/recover your Editor/Assembler option 5 PROGRAM files on cassette. This is great for people who have 32K and a cassette system but have not added disk.

Now let us take a look at RELATIVE files. They may be used on disk only. The two types of file access are RELATIVE and SEQUENTIAL. Records in a RELATIVE (random) file may be accessed in any order. An example of a RELATIVE file is the disk directory.

Try the following:

```
10 OPEN #1:"DSK1.",INPUT,INTERNAL, RELATIVE
20 INPUT A :: INPUT #1,REC A:A$
30 PRINT A$ :: GOTO 20
```

Just put in a number in A from 0 to 127, 0 (zero) will give you the disk name. 1 to 127 will give you the filename on that record. If you get a null string you are beyond the end of the directory. I use a RELATIVE file on the SPIRIT OF '99 TIBBS which contains the alphabetical user list. This makes it possible to search the list for a name. RELATIVE files must have FIXED length records. This means that any extra space in the record is filled in by the computer. If you have a 70 character line in a FIXED 80 file the last 10 characters will not be available for use. RELATIVE files may be DISPLAY or INTERNAL.

Variable files (INTERNAL or DISPLAY) may only be accessed sequentially.

DISPLAY VARIABLE 80 files are most likely text files. This article is being written in Extended BASIC

and stored as D/V 80 on disk. You may print D/V 80 files from TI-Writer (or equivalent) DM1000 or the Editor/Assembler editor. The latter editor will strip out all control characters. You may need to use the TI-Writer Formatter for some files.

If you OPEN #1:"DSK1.TEST" The file will default to D/V 80 in UPDATE mode. This means you may read or write to it. If you open it in INPUT mode you may read from it. Interestingly, if you open the file in INPUT mode the default is D/V with no record length. I find this useful if I need to read a variety of files with speech.

One seeming hitch to this is with the Horizon RAMdisk. There the INPUT default is D/V 80. The OUTPUT (write) default is D/V 80. You may also find D/V 80 files as RLE pictures or pictures which are intended to be printed directly on your printer. D/V 80 files may also be source code for assembly programs. Those may be assembled with the Editor/Assembler cartridge.

One of the most interesting files is DISPLAY VARIABLE 163. This is normally used by the computer for MERGE format. If you try to read one of these files you will get some numbers and strings but mostly unprintable characters. The computer uses tokens to represent words such as CALL COLOR, CLEAR etc.. Tokens are used to represent line numbers, quoted and unquoted strings and much more. If you see a D/V 163 file go into Extended BASIC, type NEW, type MERGE DSKx.(filename) and the file should be in memory after reading it from disk. If you wish to save it as a PROGRAM file, do so as if you had typed it in.

DISPLAY FIXED 80 files may be assembly files. They may be loaded from Editor/Assembler option 3 or from Extended BASIC with CALL LOAD. There is no way that I know of to tell the difference without trying to load the file. Note that some early versions of TEII store files in D/F 80 in terminal mode.

DISPLAY FIXED 128 files may be RLE pictures or Archived (packed) files. If you cannot tell by the name you may have to try to unpack the file with Archiver. The usual indicator for an Archiver file is /PAC or /ARC as the last part of the file name. This is not always followed however. With the advent of archiving programs that squeeze or compress the file this may become more confusing. Be sure to read any instructions descriptions etc., that come with the file. This is just an overview of some of the types of files you might run into. You will find numerous exceptions to these guide lines.

Retyped for TEXPAC BBS by John Ryan of TISHUG.

RELATIVE Files

by Kevin Cox and John Ryan

We came up with this variation of SPIRIT OF '99's Irwin Hott's program in the above article.

```
100 OPEN #1 : "DSK1.",INTERNAL,RELATIVE
110 FOR FILE=0 TO 127
120 INPUT #1,REC FILE:A$
130 PRINT A$,
140 IF A$="" THEN END
150 NEXT FILE
```

If any member is interested in another like these to show files along with their format PGM or D/V 80, drop SARA a line on the BBS.

For Sale

One TI 994/A Home Computer.

2 TI joysticks

National RQ-8100 Computer Cassette Recorder, TI Extended BASIC, Alligator Mix, Munch Man, Car Wars, Hangman, Addition and Subtraction 2, Adventure Module with manual, Pirate adventure and Voodoo Castle. TI Touch Typing Tutor and Decimals.

Also some tapes; eg Toad, Oldies but Goodies II, 18 TI Club games Tapes and 8 blank tapes.

All this for \$160 or nearest offer.

If you are interested, please ring Dick on (02)449 1773 or leave mail for CHOCKY

continued from page 1

My friend Lou Amadio was enhancing his console even further than the 64K on the 16 bit bus by putting in Extended BASIC following the article by Ross Mudie (Vol7 No7 P20). While doing so, he found some errors (origin unknown) in the paragraph at the top of the second column. The sentence starting "Pin 34", should read:

Pin 34 is *ROMG (the active low bank select line); the DPDT switch must connect pin 34 from the mother board to either the Extended BASIC module or the cartridge port connector. Two additional resistors must be provided from pin 34 of Extended BASIC to +5V and from pin 34 of the cartridge port connector to +5V. The resistor for Extended BASIC can be conveniently mounted on its PCB, whilst the resistor for the cartridge port connector can be mounted close to the pins on the solder side of the cartridge port connector, between pins 34 and 19.

The diagrams are correct so I am not sure how pin 34 became pin 29 in that description. Apologies all round.

Wanted, someone with patience and time, good at spelling and grammar, able to use a word processor, a good eye for layout, preferably two systems if other members of the family want access to the computer, space for a rather noisy printer which would not upset others if it runs for many hours into the night, at least two disk drives, preferably double sided and double density and who does not mind working long hours for no pay and precious little encouragement. If there is someone you know who might fit all or some of these requirements then you are probably kidding yourself! Seriously, in just a few months, two more issues of TND in fact, it will be AGM time and the post of Editor of TND will be available for someone else to grab. There must be some good things about the job, surely? Well, sometimes we get the Newsletters first to read but only because we do a review of them I suspect and I get to access the BBS which is informative if nothing else. I also have met more people and presumably more people know me which may be good and may be bad. I also have the use of a modem and a printer buffer which makes the use of the 30 cps printer reasonable. I was not supplied with a second disk controller which I asked for, so I was forced to buy one myself. Perhaps I was not a good enough negotiator of terms and conditions at the start of my contract. As I recall, I only asked for Editorial freedom which I have received, although some were not amused about April after the event, were they Chris. However all good things must come to an end someone said, or was it life was not meant to be easy? I may not be in the country for the first part of next year and this is my first warning that a new editor may be required. Perhaps I am wrong and there is someone eager and willing to take over. I hope so, as I believe that the Illawarra regional group has provided TISHUG with an excellent publication over the last two years and I would not like to see all our good efforts dissipate through lack of interest.

For the last two AGMs I have prepared documents which I hoped would cause some discussion about the future direction of TISHUG and improvements that could be made to encourage members to remain as members. Unfortunately the members who attended the AGM did not see that there was anything to worry about so I am starting again a bit earlier to allow some correspondence from remote members to be received, if they feel like putting pen to paper or word processor to printer about problems with TISHUG as they see them. The question I think we should address is: what is the purpose of a group like TISHUG? I know what it says in the articles of association, a lot of arm waving and do gooding everywhere. What does the member in Sydney get or the member in Darwin get and what could they get from membership in TISHUG? I think that there are basically two things that a group like ours can and should provide to all its members equally, information and software. These are the things that are almost impossible for an individual to get from any other source except by a huge effort, and also the things that are easily attracted to a large group. The

problem is that once attracted it is very difficult to get the software and information out of the pile at the centre of the group to the people who are not in the inner group.

For example, Terry has all the club software at his home I assume. Well someone has to have it but that means that that member is more privileged than all others. He works for that privilege of course, but all those who live close to him, or see him regularly can share in that privilege to a certain extent while those who live at Wagga or Coffs Harbour have much more difficulty in sharing. The same thing applies to the Newsletters which are our main source of external information. In fact I believe that Tony McGovern of Funnelweb fame was a member of TISHUG who objected to the inability of a member who lived in Newcastle to gain access to the latest information to help him use his computer to its utmost. Perhaps the TI99/4A community will praise TISHUG for having that attitude and thus forcing the creation of the Hunter Valley group and Funnelweb Farm. In terms of keeping good members in TISHUG it was a disaster however. I believe that both Hunter Valley and Brisbane have methods for circulating all the information that comes into their groups to all their members. They also provide software at a much more reasonable price than TISHUG. How can Brisbane sell software at \$2 per disk to a member in NSW while the same disk from TISHUG is \$5? Is this what the members of TISHUG want? In fact it does not seem to be what the authors of Freeware or Shareware software want to happen to their product. Think about it for a month and if you want to have a say, send a letter to the directors or use the BBS. If you want your views published, send them to EDITOR on the BBS, or to 20 Robsons Road, Keiraville, NSW 2500.

I am preparing this TND with a console with 64K of RAM on the 16 bit bus (only using 32K of it) and with my 512K RAMdisk. I am very pleased with it as the response to the keyboard is fast enough so that no characters are missed by TI-Writer. It is probably faster in edit commands as well and all the little beeps that Funnelweb puts out are shorter than before. Multiplan is also significantly faster when on input and changing from alpha/value on the first key press. I have set my RAMdisk up with 3 pseudo disks. Two of them are just big enough for Multiplan (called TIMP) and the two dictionaries of the spelling check program (called DICT). Multiplan looks for its files to a disk called TIMP and if the RAMdisk is at CRU address >1000, then it finds its files on the RAMdisk very quickly. I had to modify the Spelling check program to look for its dictionaries as DSK.DICT.1 and DSK.DICT.2. The RAMdisk and the speedy memory has made the spelling check a reasonable thing to do. A large file (>30 sectors) takes about 2 minutes to read in and process the dictionaries. The main disk on the RAMdisk contains the menu program and Funnelweb with a number of other programs. With the normal menu and all of Funnelweb's menus quite a lot can be done with only a few keystrokes. It can all be personalised quite easily as well. I do not think that a PC offers such a flexible system with the access speed of the TI99/4A, even with a hard disk. Of course the programs may run a touch slower, but they do their job with few bugs. I am trying to get enough time to write a few programs in C, which is where I was a month ago when I was trying to get my 512K RAMdisk working. It had a problem with the 5 volt regulator which destroyed 4 8K RAM chips and then a problem with a bad connection in a socket.

I have only 3 newsletters to tell you about this month as Terry was away and the others did not reach me. The first is a welcome return from Channel 99 users in Hamilton Ontario. I visited them last year and was upset that they had some problems for a while. This is the September issue and the club has been rather quiet in their summer. There is a review of the Rave99 keyboard and of EZkeys, an Extended BASIC program to view a DV80 file and a report on a Telco update (V2.1).

The September issue from Ottawa has a review of Spad XIII, a letter to Fairware authors telling them how best to tell us, the consumers, what they want from us, a quick tutorial on AUTOEXEC for MDOS, a review on

continued on page 18

Regional Group Reports

Meeting summary.

Banana Coast	13/11/88 Sawtell
Carlingford	16/11/88 Carlingford
Central Coast	12/11/88 Toukley
Glebe	10/11/88 Glebe
Illawarra	20/11/88 Keiraville
Liverpool	11/11/88 ???
Northern Suburbs	??/11/88 Davidson
Sutherland	18/11/88 Jannali

BANANA COAST Regional Group (Coffs Harbour area)

Regular meetings are held in the Sawtell Tennis Club on the second Sunday of the month at 2 pm sharp. For information on meetings of the Banana Coast group, contact Kevin Cox at 7 Dewing Close, Bayldon, telephone (066)53 2649, or John Ryan of Mullaway via the BBS, user name SARA, or telephone (066)54 1451.

The September meeting of our group was held at our usual venue, with 7 members attending. John Ryan demonstrated the Business Graphs Disk to all, filling in the local rain fall as data for the graphs. John also supplied the latest information from the BBS. After a hearty afternoon tea, the meeting concluded with a few minor queries on selected programs. At our last meeting we had the pleasure of Bob Bishop from the Melbourne Group attendind. He provided us with a wealth of knowledge and entertainment.

Regards, Kevin Cox.

CARLINGFORD Regional Group.

Regular meetings are usually on the third Wednesday of each month at 7.30pm. Contact Chris Buttner, 79 Jenkins Rd, Carlingford, (02)871 7753, for more information.

CENTRAL COAST Regional Group.

Meetings are normally held on the second Saturday of each month, 6.30pm at the Toukley Tennis Club hall, Header St, Toukley. Contact Russell Welham (043)92 4000

GLEBE Regional Group.

Regular meetings are normally on the Thursday evening following the first Saturday of the month, at 8pm at 43 Boyce St, Glebe. Contact Mike Slattery, (02)692 0559.

ILLAWARRA Regional Group.

Regular meetings are normally on the third Monday of each month, except January, at 7.30pm, Keiraville Public School, Gipps Rd, Keiraville, opposite the Keiraville shopping centre. Contact Bob Montgomery on (042)28 6463 for more information.

LIVERPOOL Regional Group

Regular meeting date is the Friday following the TISHUG Sydney meeting at 7.30 pm. Contact Larry Saunders (02)644 7377 (home) or (02)759 8441 (work) for more information.

Meetings coming up.

NORTHERN SUBURBS Regional Group.

If you want any information please ring Dennis Norman on (02)452 3920, or Dick Warburton on (02)918 8132.

SUTHERLAND Regional Group.

Regular meetings are held on the third Friday of each month at the home of Peter Young at Jannali at 7.30pm. Group co-ordinator is Peter Young, (02) 528 8775. BBS Contact is Gary Wilson, user name VK2YGW on this BBS.

TISHUG in Sydney

Regular meetings are normally at 2pm on the first Saturday the month, except January and possibly other months with public holidays on that weekend, at the Woodstock Community Centre, Church Street, Burwood.

TISHUG (Aust) Limited - Main meetings.

November 5 - Full day tutorial workshop.

A run down of activities on this day will include tutorials by Craig Sheehan on his XDP utility routines, demonstrations by Ross Mudie on his peripheral control device, selected software demonstrations by Russell Welham and disk copying by Terry Phillips. In addition there will be normal shop and library facilities.

Lunch will be available, a hamburger and soft drink for the modest cost of \$2. Volunteers are required to attend to the BBQ and do some cooking etc. If you wish to volunteer, see me on the day.

Set up at Woodstock will be at 9am, with a scheduled start of 10am.

November 6 (Sunday) - Hunter Valley amusement day.

Contact Albert Anderson if you are going on (046)66 2602.

December 3 - Christmas party.

Given a fine day this will be one of the major events of the year, with plenty of food and drink available and a great chance to chat with fellow members in a social relaxed environment. There will be plenty of software released for this meeting so you will have plenty to keep you occupied over the Christmas/New Year holiday break.

January - no meeting scheduled.

This is the month when we can all relax and enjoy the summer break.

February 4 (1989) - Annual General Meeting.

Providing the club auditorium is available, then this meeting will be held at the Burwood RSL Club, Shaftsbury Road, Burwood. This meeting will see the election of a board of Directors to run the group for the following 12 months. Full details and nomination forms will appear in the December issue of the news digest.

Enquiries on information contained in this file can be directed to the Secretary through the following media:

BBS - username SECRETARY
Voice - telephone (02)797 6313
Mail - Secretary
TISHUG (Aust) Ltd
PO Box 214
Redfern NSW 2016

continued from page 27

The Forth Column

The Forum

Thought for the month:

"The attention span of a computer is only as long as its electrical cord"

Back when I was first learning about Forth, Lee Stewart was writing words that impressed and inspired me to learn this "new" language. One of the few shortcomings was the lack of speech. Lee came up with a word to take care of that. This word takes a number off the stack which corresponds to the ROM location of the word to be said, and speaks that word. The list of the words and their locations can be found on pages 422 to 427 of the Editor/Assembler manual.

: SAY (n ---) 4 0 DO 4 SRC DUP FOOO AND 4 SRL
4000 + 9400 ! LOOP
4000 9400 ! 5000 9400 ! DROP ; (This word is in HEX)

I have this word on screen #3 of my system disk and I follow it with:

56B3 3A32 2D19 3793 6551 SAY SAY SAY SAY SAY

Have fun playing around with this word and do not forget to experiment! What happens if you specify a location where a defined word does not exist? Go ahead and try it and I will see you next month.

George L. Smyth
3017 Sylvan Drive
Falls Church, VA 22042
(703)533-8710